

Constant Unary Constraints and Symmetric Real-Weighted Counting Constraint Satisfaction Problems*

TOMOYUKI YAMAKAMI†

Abstract: A unary constraint (on the Boolean domain) is a function from $\{0, 1\}$ to the set of real numbers. A free use of auxiliary unary constraints given besides input instances has proven to be useful in establishing a complete classification of the computational complexity of approximately solving weighted counting Boolean constraint satisfaction problems (or #CSPs). In particular, two special “constant” unary constraints are a key to an arity reduction of arbitrary constraints, necessary for the classification. To clarify the essential role of auxiliary free unary constraints, we demonstrate the efficient approximability of the two constant unary constraints by an arbitrary nonempty set of real-valued constraints. By a direct application of this approximability result, we construct polynomial-time randomized approximation-preserving Turing reductions (or AP-reductions) to any given #CSP composed of symmetric real-valued constraints of arbitrary arities from #CSP(g) for an appropriate constraint g without any use of extra unary constraints. Moreover, we can give a precise description of this constraint g .

Keywords: counting constraint satisfaction problem, AP-reducible, T-constructible, constant unary constraint, symmetric constraint, algebraic real number, p-convergence

1 Roles of Constant Unary Constraints

Constraint satisfaction problems (or *CSPs*, in short) are combinatorial problems that have been ubiquitously found in real-life situations. The importance of these problems have led recent intensive studies from various aspects: for instance, decision CSPs [5, 9], optimization CSPs [3, 13], and counting CSPs [1, 4, 7, 11]. Driven by theoretical and practical interests, in this paper, we are particularly focused on *counting CSPs* (abbreviated as #CSPs) whose goal is to count the number of variable assignments that satisfy all given Boolean constraints defined over a fixed series of Boolean variables. The problem of counting the number of Boolean assignments that satisfy each given propositional formula, known as #SAT (counting satisfiability problem), is a typical counting CSP with Boolean-valued constraints, *AND*, *OR*, and *NOT*. As this example demonstrates, in most real-life applications, all available constraints are pre-determined, we naturally fix a collection of “allowed” constraints, say, \mathcal{F} and wish to compute each solution of any #CSP whose constraints are all chosen from \mathcal{F} . Such a problem is conventionally denoted #CSP(\mathcal{F}) and this notation will be used throughout this paper. Creignou and Hermann [4] first examined the computational complexity of exactly counting the solutions of unweighted #CSPs. Recently, Dyer, Goldberg, and Jerrum [8] studied the computational complexity of approximately computing the solutions of unweighted #CSPs using a technical approximate reduction, known as *polynomial-time randomized approximation-preserving Turing reduction* (or *AP-reduction*, hereafter), whose formulation is originated from [6] and it will be explained in Section 2.2.

In the case of *weighted #CSPs*, the values of constraints are expanded from Boolean values to more general values, and the goal of each weighted #CSP is to calculate the sum, over all possible assignments for Boolean variables, of products of the output values of all given constraints. By further allowing a free use of auxiliary unary constraints besides initially given input constraints, Cai, Lu, and Xia [2] pioneered a study on a classification of the complexity of *exactly* solving complex-weighted #CSPs restricted on a given set \mathcal{F} of constraints. Similarly, in the presence of auxiliary unary constraints, a classification of the complexity of *approximately* solving complex-weighted #CSPs was presented first in [11]. In this classification, the free use of auxiliary unary constraints provide enormous power that makes it possible to obtain a “dichotomy” theorem rather than a “trichotomy” theorem of Dyer et al. [8] for Boolean-valued constraints (or simply, *Boolean constraints*). Limited to unweighted #CSPs, a key to the proof of the trichotomy theorem in [8] is an efficient approximation of so-called *constant unary constraints*, conventionally denoted[‡] $\Delta_0 = [1, 0]$ and $\Delta_1 = [0, 1]$. This efficient approximation makes us first analyze the complexity of #CSP(\mathcal{F}) using those

*A preliminary version under a slightly concise title appeared in the Proceedings of the 23rd International Symposium on Algorithms and Computation (ISAAC 2012), Taipei, Taiwan, December 19–21, 2012, Lecture Notes in Computer Science, Springer-Verlag, vol.7676, pp.237–246, 2012.

†Present Affiliation: Department of Information Science, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan

‡A bracket notation $[x, y]$ denotes a unary function g satisfying $g(0) = x$ and $g(1) = y$. Similarly, $[x, y, z]$ expresses a binary function g for which $g(0, 0) = x$, $g(0, 1) = g(1, 0) = y$, and $g(1, 1) = z$.

constant unary constraints and then eliminate them entirely to obtain a desired classification theorem. In this paper, we claim that, even in the case of weighted $\#CSP$ s, more specifically, whenever \mathcal{F} is composed of real-valued constraints, at least one of Δ_0 and Δ_1 is always approximated efficiently from \mathcal{F} . This weaker approximation result turns out to be sufficient to demonstrate the desired classification theorem.

Theorem 1.1 *For any nonempty set \mathcal{F} of real-valued constraints, there exists a constant unary constraint $h \in \{\Delta_0, \Delta_1\}$ for which $\#CSP(h, \mathcal{F})$ is AP-equivalent to $\#CSP(\mathcal{F})$ (i.e., $\#CSP(h, \mathcal{F})$ is AP-reducible to $\#CSP(\mathcal{F})$ and vice versa).*

When the values of constraints in \mathcal{F} are all limited to Boolean values, Theorem 1.1 was already proven in [8] based on basic properties of Boolean arithmetic. For real-valued constraints, however, we cannot rely on those properties and thus we need to develop a quite different argument. An important ingredient of our proof is an efficient estimation of a lower bound of an arbitrary polynomial in the values of given constraints. However, since our constraints can output negative real values, the polynomial may possibly produce arbitrary small values that we cannot find a polynomial-time computable lower bound. To cope with such a difficult situation, we inevitably restrict our attention onto *algebraic real numbers*.

With Theorem 1.1 as a technical tool, we are able to demonstrate an approximation classification of real-weighted $\#CSP$ s when arbitrary free unary constraints are permitted to assist standard inputs since the constant unary constraints are quite valuable in “reducing” constraints of high arity to those of low arity. In an exact counting model, both Δ_0 and Δ_1 are naturally available to use; however, in our approximate counting model, we rely on Theorem 1.1 and it guarantees only the availability of either Δ_0 or Δ_1 . Even with a help of a single constant unary constraint, it is still possible to reduce the arities of target constraints. Furthermore, we can build the reduction with no use of auxiliary unary constraint.

More precisely, let us denote by \mathcal{U} the set of all unary constraints. Given an arbitrary constraint f , the free use of auxiliary unary constraints makes $\#SAT_{\mathbb{C}}$ (a complex extension of $\#SAT$) AP-reducible to $\#CSP(f, \mathcal{U})$ unless f is factored into three categories of constraints: the binary equality, the binary disequality, and unary constraints [11]. All constraints factored into constraints in those categories form a special set \mathcal{ED} . The aforementioned fact establishes the following complete classification of the approximation complexity of weighted $\#CSP$ s in the presence of \mathcal{U} .

Theorem 1.2 [11, Theorem 1.1] *Let \mathcal{F} be any set of complex-valued constraints. If $\mathcal{F} \subseteq \mathcal{ED}$, then $\#CSP(\mathcal{F}, \mathcal{U})$ is solvable in polynomial time; otherwise, it is AP-reduced from $\#SAT_{\mathbb{C}}$.*

The proof of this dichotomy theorem in [11] employed two technical notions of “factorization” and “T-constructibility” of constraints. Because the proof is rather complicated and thus lengthy, it is not immediately clear what roles the auxiliary free unary constraints play in the proof of this classification theorem. Toward a complete classification of the approximation complexity of $\#CSP$ s *without* any auxiliary unary constraint, it is therefore beneficial to clarify the roles of those extra constraints. With a careful application of Theorem 1.1, we can precisely locate a point where the free auxiliary unary constraints are imperatively required in establishing the desired dichotomy theorem. For this purpose, we wish to present a new alternative proof in which the use of the auxiliary unary constraints is made only at the very end of the proof. To simplify our argument further, we intend to restrict our attention only on *symmetric real-weighted $\#CSP$ s* in the subsequent sections.

Our alternative proof is outlined as follows. In the first step, we must recognize constraints g of the following special forms: $[0, x, z]$ with $x, y > 0$ and $[x, y, z]$ with $x, y, z \neq 0$ and $xz \neq y^2$. Those constraints g become crucial elements of our later analysis because, when auxiliary unary constraints are available for free, $\#CSP(g, \mathcal{U})$ is known to be computationally at least as hard as $\#SAT$ with respect to AP-reducibility by Theorem 1.2. In the second step, we must isolate a set \mathcal{F} of constraints whose corresponding counting problem $\#CSP(\mathcal{F})$ is AP-reduced from a certain $\#CSP(g)$ with no use of the auxiliary unary constraints. To be more exact, we wish to establish the following specific AP-reduction between $\#CSP(\mathcal{F})$ and $\#CSP(g)$.

Theorem 1.3 *Let \mathcal{F} be any set of symmetric real-valued constraints of arity at least 2. If either $\mathcal{F} \subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$ or $\mathcal{F} \subseteq \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$ holds, then $\#CSP(\mathcal{F})$ is polynomial-time solvable. Otherwise, $\#CSP(\mathcal{F})$ is AP-reduced from $\#CSP(g)$, where g is an appropriate constraint of the special form described above.*

In this theorem, the constraint set \mathcal{DG} consists of *degenerate* constraints, \mathcal{ED}_1 indicates a “generalized” \mathcal{ED} , and \mathcal{AZ} contains specific symmetric constraints having alternating zeros. Two additional sets $\mathcal{DG}^{(-)}$ and $\mathcal{ED}_1^{(+)}$ are naturally induced from \mathcal{DG} and \mathcal{ED}_1 , respectively. For their precise definitions, refer to

Section 4.

Although $\#\text{CSP}(\mathcal{DG} \cup \mathcal{ED}_1^{(+)})$ and $\#\text{CSP}(\mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ})$ are both solvable in polynomial time (Proposition 4.2), they behave quite differently in the presence of auxiliary free unary constraints. The counting problem $\#\text{CSP}(\mathcal{DG} \cup \mathcal{ED}_1^{(+)}, \mathcal{U})$ remains solvable in polynomial time; on the contrary, $\#\text{CSP}(\mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}, \mathcal{U})$ is AP-reduced from $\#\text{SAT}$ (as a consequence of Claim 5). Since these facts immediately prove Theorem 1.2 for symmetric real-weighted $\#\text{CSPs}$, the aforementioned proof outline gives an alternative proof to Theorem 1.2, in which the use of auxiliary free constraints is limited to the very end of the proof.

Comparison of Proof Techniques: In [8], the approximation of the constant unary constraints by any set of Boolean constraints was proven with a notion of “simulatability.” Instead, our proof of Theorem 1.1 employs a direct arity reduction and applies an estimation result in [11]. While a key proof technique used to prove Theorem 1.2 is the factorization of constraints, our proof of Theorem 1.3 (which leads to Theorem 1.2) makes a heavy use of the constant unary constraints. This fact makes the proof cleaner and more straightforward to follow.

2 Fundamental Notions and Notations

We will explain basic concepts that are necessary to read through the rest of this paper. First, let \mathbb{N} denote the set of all *natural numbers* (i.e., nonnegative integers) and let \mathbb{R} be the set of all *real numbers*. For convenience, define $\mathbb{N}^+ = \mathbb{N} - \{0\}$ and, for each number $n \in \mathbb{N}^+$, $[n]$ stands for the integer interval $\{1, 2, \dots, n\}$.

Because our results heavily rely on Lemma 2.1(3), we need to limit our attention within *algebraic real numbers*. For this purpose, a special notation \mathbb{A} will be used to indicate the set of all algebraic real numbers. To simplify our terminology throughout the paper, whenever we refer to “real numbers,” we actually mean “algebraic real numbers” as long as there is no confusion from the context.

2.1 Constraints and $\#\text{CSPs}$

The term “constraint of arity k ” always refers to a function mapping the set $\{0, 1\}^k$ of binary strings of length k to \mathbb{A} . Assuming a standard lexicographic order on the set $\{0, 1\}^k$, we conveniently express f as a *row-vector* consisting of its output values; for instance, when f has arity 2, it is expressed as $(f(00), f(01), f(10), f(11))$. Given any k -ary constraint $f = (f_1, f_2, \dots, f_{2^k})$ in a vector form, the notation $\|f\|_\infty$ means $\max_{i \in [2^k]} \{f_i\}$. A k -ary constraint f is called *symmetric* if, for every input x in $\{0, 1\}^k$, the value $f(x)$ depends only on the Hamming weight (i.e., the number of 1’s) of the input x ; otherwise, f is called *asymmetric*. For any symmetric constraint f of arity k , we also use a succinct notation $[f_0, f_1, \dots, f_k]$ to express f , where each entry f_i expresses the value of f on inputs of Hamming weight i . For instance, if $f = [f_0, f_1, f_2]$ is of arity two, then it holds that $f_0 = f(00)$, $f_1 = f(01) = f(10)$, and $f_2 = f(11)$. Of all symmetric constraints, we recognize two special unary constraints, $\Delta_0 = [1, 0]$ and $\Delta_1 = [0, 1]$, which are called *constant unary constraints*.

Restricted to a set \mathcal{F} of constraints, a *real-weighted (Boolean) $\#\text{CSP}$* , conventionally denoted $\#\text{CSP}(\mathcal{F})$, takes a *finite* set Ω composed of elements of the form $\langle h, (x_{i_1}, x_{i_2}, \dots, x_{i_k}) \rangle$, where $h \in \mathcal{F}$ is a function on k Boolean variables $x_{i_1}, x_{i_2}, \dots, x_{i_k}$ in $X = \{x_1, x_2, \dots, x_n\}$ with $i_1, \dots, i_k \in [n]$, and its goal is to compute the real value

$$csp_\Omega =_{\text{def}} \sum_{x_1, x_2, \dots, x_n \in \{0, 1\}^n} \prod_{\langle h, (x_{i_1}, x_{i_2}, \dots, x_{i_k}) \rangle \in \Omega} h(x_{i_1}, x_{i_2}, \dots, x_{i_k}),$$

where x denotes $(x_{i_1}, x_{i_2}, \dots, x_{i_k})$. To illustrate Ω graphically, we tend to view it as a *labeled undirected bipartite graph* $G = (V_1 | V_2, E)$ whose nodes in V_1 are labeled distinctively by x_1, x_2, \dots, x_n in X and nodes in V_2 are labeled by constraints h in \mathcal{F} such that, for each pair $\langle h, (x_{i_1}, x_{i_2}, \dots, x_{i_k}) \rangle$, there are k edges between an associated node labeled h and the nodes labeled $x_{i_1}, x_{i_2}, \dots, x_{i_k}$. The labels of nodes are formally specified by a *labeling function* $\pi : V_1 \cup V_2 \rightarrow X \cup \mathcal{F}$ with $\pi(V_1) \subseteq X$ and $\pi(V_2) \subseteq \mathcal{F}$ but we often omit it from the description of G for simplicity. When Ω is viewed as this bipartite graph, it is called a *constraint frame* [11, 12]. More formally, a constraint frame $\Omega = (G, X | \mathcal{F}', \pi)$ is composed of an undirected bipartite graph G with its associated labeling function $\pi : V_1 \cup V_2 \rightarrow X \cup \mathcal{F}'$, a variable set $X = \{x_1, x_2, \dots, x_n\}$, and a *finite* set $\mathcal{F}' \subseteq \mathcal{F}$.

To simplify later descriptions, we wish to use the following simple rule of abbreviation. For instance, when f is a constraint and both \mathcal{F} and \mathcal{G} are constraint sets, we write $\#\text{CSP}(f, \mathcal{F}, \mathcal{G})$ to mean $\#\text{CSP}(\{f\} \cup \mathcal{F} \cup \mathcal{G})$.

In the subsequent sections, we will use the following succinct notations. Let f be any constraint of arity $k \in \mathbb{N}^+$. Given any index $i \in [k]$ and any bit $c \in \{0, 1\}$, the notation $f^{x_i=c}$ stands for the function g of arity $k-1$ satisfying that $g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) = f(x_1, \dots, x_{i-1}, c, x_{i+1}, \dots, x_k)$ for every $(k-1)$ -tuple $(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) \in \{0, 1\}^{k-1}$. For any two distinct indices $i, j \in [k]$, we denote by $f^{x_i=x_j}$ the function g defined as $g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) = f(x_1, \dots, x_{i-1}, x_j, x_{i+1}, \dots, x_k)$ for every k -tuple $(x_1, x_2, \dots, x_k) \in \{0, 1\}^k$. Finally, let $f^{x_i=*}$ express the function g defined by $g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) = \sum_{c \in \{0, 1\}} f(x_1, \dots, x_{i-1}, c, x_{i+1}, \dots, x_k)$ for every $(k-1)$ -tuple $(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k) \in \{0, 1\}^{k-1}$.

2.2 $\text{FP}_{\mathbb{A}}$ and AP-Reducibility

To connect our results (particularly, Theorems 1.1 and 1.3) to Theorem 1.2, we will follow notational conventions used in [11, 12]. First, $\text{FP}_{\mathbb{A}}$ denotes the collection of all \mathbb{A} -valued functions that can be computed deterministically in polynomial time.

Let F be any function mapping $\{0, 1\}^*$ to \mathbb{A} and let Σ be any nonempty finite alphabet. A *randomized approximation scheme* (or RAS, in short) for F is a randomized algorithm that takes a standard input $x \in \Sigma^*$ together with an error tolerance parameter $\varepsilon \in (0, 1)$, and outputs values w with probability at least $3/4$ for which

$$\min\{2^{-\varepsilon}F(x), 2^{\varepsilon}F(x)\} \leq w \leq \max\{2^{-\varepsilon}F(x), 2^{\varepsilon}F(x)\}. \quad (1)$$

Given two arbitrary real-valued functions F and G , a *polynomial-time randomized approximation-preserving Turing reduction* (or *AP-reduction*) from F to G [6] is a randomized algorithm M that takes a pair $(x, \varepsilon) \in \Sigma^* \times (0, 1)$ as input, uses an arbitrary RAS N for G as oracle, and satisfies the following three conditions: (i) with an arbitrary RAS N as oracle, M is always an RAS for F ; (ii) every oracle call made by M is of the form $(w, \delta) \in \Sigma^* \times (0, 1)$ with $1/\delta \leq \text{poly}(|x|, 1/\varepsilon)$ and its answer is the outcome of N on (w, δ) ; and (iii) the running time of M is upper-bounded by a certain polynomial in $(|x|, 1/\varepsilon)$, which is not dependent of the choice of N . If such an AP-reduction exists, then we also say that F is *AP-reducible* to G and we write $F \leq_{\text{AP}} G$. If both $F \leq_{\text{AP}} G$ and $G \leq_{\text{AP}} F$ hold, then F and G are said to be *AP-equivalent* and we use the special notation $F \equiv_{\text{AP}} G$.

2.3 Effective T-Constructibility

Our goal in the subsequent sections is to prove our main theorems, Theorems 1.1 and 1.3. For their desired proofs, we will introduce a fundamental notion of *effective T-constructibility*, whose underlying idea is borrowed from a graph-theoretical formulation of *limited T-constructibility* in [12].

Let f be any constraint. We say that an undirected bipartite graph $G = (V_1 | V_2, E)$ (together with a labeling function π) *represents* f if V_1 consists only of k nodes labeled x_1, \dots, x_k , which may possibly have a certain number of dangling[§] edges, and V_2 contains only a node labeled f to whom each node x_i is adjacent. Given a set \mathcal{G} of constraints, a graph $G = (V_1 | V_2, E)$ is said to *realize* f by \mathcal{G} if the following four conditions are met simultaneously:

- (i) $\pi(V_2) \subseteq \mathcal{G}$,
- (ii) G contains at least k nodes having the labels x_1, \dots, x_k , possibly together with nodes associated with other variables, say, y_1, \dots, y_m ; namely, $V_1 = \{x_1, \dots, x_k, y_1, \dots, y_m\}$,
- (iii) only the nodes x_1, \dots, x_k are allowed to have dangling edges, and
- (iv) $f(x_1, \dots, x_k) = \lambda \sum_{y_1, \dots, y_m \in \{0, 1\}} \prod_{w \in V_2} f_w(z_1, \dots, z_d)$ for an appropriate constant $\lambda \in \mathbb{A} - \{0\}$, where f_w denotes a constraint $\pi(w)$ and $z_1, \dots, z_d \in V_1$.

Furthermore, we say that f is *effectively T-constructible* from \mathcal{G} if, for any integer $m \geq 2$ and for any graph G representing f with distinct variables x_1, \dots, x_k , there always exists another graph G' satisfying the following two conditions:

- (i') G' realizes f by \mathcal{G} , and
- (ii') G' maintains the same dangling edges as G does.

When f is effectively T-constructible from \mathcal{G} , we write $f \leq_{\text{e-con}} \mathcal{G}$. When \mathcal{G} is a singleton $\{g\}$, we succinctly write $f \leq_{\text{e-con}} g$.

[§]A *dangling edge* is obtained from an edge by deleting exactly one end of the edge. These dangling edges are treated as “normal” edges, and therefore the degree of each node counts dangling edges as well.

We define a *sign function*, denoted sgn , as follows. For any real number λ , we set $\text{sgn}(\lambda) = +1$ if $\lambda > 0$, $\text{sgn}(\lambda) = 0$ if $\lambda = 0$, and $\text{sgn}(\lambda) = -1$ if $\lambda < 0$.

An infinite series $\Lambda = (g_1, g_2, g_3, \dots)$ of arity- k constraints is called a *p-convergence series*[¶] for a target constraint $f = (x_1, x_2, \dots, x_{2k})$ of arity k if there exist a constant $\lambda \in (0, 1)$ and a deterministic Turing machine (or a DTM, in short) M running in polynomial time such that, for every number $m \in \mathbb{N}^+$, (i) M takes an input of the form 1^m and outputs the complete description of the constraint g_m in a row-vector form $(z_1, z_2, \dots, z_{2k})$, (ii) for every k -tuple $x \in \{0, 1\}^k$, if $f(x) \neq 0$, then $\text{sgn}(f(x)) = \text{sgn}(g_m(x))$, and (iii) for every index $i \in [2^k]$, if $x_i \neq 0$, then

$$\min\{(1 + \lambda^m)z_i, (1 - \lambda^m)z_i\} \leq x_i \leq \max\{(1 + \lambda^m)z_i, (1 - \lambda^m)z_i\}, \quad (2)$$

and otherwise, $|z_i| \leq \lambda^m$. A p-convergence series $\Lambda = (f_1, f_2, \dots)$ of arity- k constraints is said to be *effectively T-constructible* from a finite set $\mathcal{G} = \{g_1, g_2, \dots, g_d\}$ of constraints (denoted $\Lambda \leq_{e\text{-con}} \mathcal{G}$) if there exists a polynomial-time DTM M such that, for every number $m \in \mathbb{N}^+$, M takes an input of the form $(1^m, G, (g_1, g_2, \dots, g_d))$, where G represents f_m with distinct variables x_1, \dots, x_k , and then M outputs a bipartite graph G_m such that (i') G_m realizes f_m by \mathcal{G} and (ii') G_m maintains the same dangling edges as G does. In this case, we also say that M *effectively T-constructs* Λ from \mathcal{G} .

Lemma 2.1 *Let f and g be arbitrary constraints and let \mathcal{F} and \mathcal{G} be arbitrary constraint sets. Let $\Lambda = (g_1, g_2, \dots)$ be any p-convergence series for f .*

1. *It holds that $f \leq_{e\text{-con}} f$ and that $f \leq_{e\text{-con}} g$ and $g \leq_{e\text{-con}} h$ imply $f \leq_{e\text{-con}} h$.*
2. *If $f \leq_{e\text{-con}} \mathcal{G}$, then $\#\text{CSP}(f, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\mathcal{G}, \mathcal{F})$.*
3. *If $\Lambda \leq_{e\text{-con}} \mathcal{G}$ and \mathcal{G} is finite, then $\#\text{CSP}(f, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\Lambda, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\mathcal{G}, \mathcal{F})$.*

It is important to note that, since constraints can output negative values, the use of *algebraic real numbers* may be necessary in the proof of Lemma 2.1(3) because the proof heavily relies on a lower bound estimation of arbitrary polynomials over algebraic numbers.

For the readability, we postpone the proof of Lemma 2.1 until Appendix.

3 Approximation of the Constant Unary Constraints

Let us prove our first main theorem—Theorem 1.1—which states that, given an arbitrary set of constraints, we can efficiently approximate at least one of the two constant unary constraints. The theorem thus makes us utilize such a constant unary constraint freely for a further analysis of constraints in Section 4.

3.1 Notion of Complement Stability

To prove Theorem 1.1, we will first introduce two useful notions regarding a certain “symmetric” nature of a given constraint. A k -ary constraint f is said to be *complement invariant* if $f(x_1, \dots, x_k) = f(x_1 \oplus 1, \dots, x_k \oplus 1)$ holds for every input tuple $(x_1, \dots, x_k) \in \{0, 1\}^k$, where the notation \oplus means the (*bitwise*) *XOR*. In contrast, we say that f is *complement anti-invariant* if, for every input $(x_1, \dots, x_k) \in \{0, 1\}^k$, $f(x_1, \dots, x_k) = -f(x_1 \oplus 1, \dots, x_k \oplus 1)$ holds. For instance, $f = [1, 1]$ is complement invariant and $f' = [1, 0, -1]$ is complement anti-invariant. In addition, we say that f is *complement stable* if f is either complement invariant or complement anti-invariant. A constraint set \mathcal{F} is *complement stable* if every constraint in \mathcal{F} is complement stable. In case that f (resp., \mathcal{F}) is not complement stable, we conveniently call it *complement unstable*.

We will split Theorem 1.1 into two separate statements, as shown in Lemma 3.1, depending on whether or not a given nonempty set \mathcal{F} of constraints is complement stable.

Lemma 3.1 *Let \mathcal{F} be any nonempty set of constraints.*

1. *If \mathcal{F} is complement stable, then $\#\text{CSP}(\Delta_i, \mathcal{F}) \equiv_{\text{AP}} \#\text{CSP}(\mathcal{F})$ holds for every index $i \in \{0, 1\}$.*
2. *If \mathcal{F} is complement unstable, then there exists an index $i \in \{0, 1\}$ for which $\#\text{CSP}(\Delta_i, \mathcal{F}) \equiv_{\text{AP}} \#\text{CSP}(\mathcal{F})$.*

[¶]At a quick glance, the approximation scheme of Eq.(2) appears quite different from that of Eq.(1). However, by setting $\varepsilon = \lambda^m$, the value $1 + \varepsilon$ approximately equals 2^ε and $1 - \varepsilon$ is also close to $2^{-\varepsilon}$ for any sufficiently large number m .

Notice that Lemma 3.1 implies Theorem 1.1. Lemma 3.1(1) can be proven rather easily, as presented below, whereas Lemma 3.1(2) requires a slightly more complicated argument.

Proof of Lemma 3.1(1). In the following proof, we will deal only with Δ_0 , because the other case is similarly handled. Let \mathcal{F} be any nonempty set of constraints and take any input instance Ω , in the form of constraint frame (G, \mathcal{F}', π) with $\mathcal{F}' \subseteq \mathcal{F}$, given to the counting problem $\#\text{CSP}(\Delta_0, \mathcal{F})$.

To simplify our proof, we will modify Ω as follows. We first merge all variable nodes (i.e., nodes with “variable” labels) adjacent to nodes labeled Δ_0 into a single node having a fresh variable label. If there are more than one adjacent nodes with the label Δ_0 , then we delete all those nodes except for one node. After this modification, we always assume that there is exactly one node, say, v_0 whose label is Δ_0 . Now, let v_1 be a unique node adjacent to v_0 and let x_0 be its variable label.

Let m denote the total number of nodes in Ω whose labels are constraints that are complement anti-invariant in \mathcal{F} . By simply removing the node labeled Δ_0 from Ω , we then obtain another instance, say, Ω' , which is obviously an input instance to $\#\text{CSP}(\mathcal{F})$. Because of properties of complement anti-invariance, the following claim holds:

$$csp_{\Omega'} = csp_{\Omega} + (-1)^m csp_{\Omega}. \quad (3)$$

Now, we want to prove Eq.(3). Let us consider any assignment σ to all variables appearing in Ω' except for x_0 . Associated with σ , we will introduce two corresponding assignments, σ_0 and $\sigma - 1$, for Ω' . Firstly, we obtain σ_0 from σ by additionally assigning 0 to x_0 . Now, we assume that σ_0 is an satisfying assignment for Ω' . Secondly, let σ_1 be defined by assigning 1 to x_0 and $1 - \sigma(z)$ to all the other variables z . For this σ_1 , the product of all constraints' values given by σ_1 equals $(-1)^m$ times the product of all constraints' values given by σ_0 . This completes the proof of Eq.(3).

In the case where m is even, we immediately obtain the equation $csp_{\Omega} = \frac{1}{2}csp_{\Omega'}$ from Eq.(3). Now, assume that m is odd. In this case, we choose a constraint g that is complement anti-invariant in \mathcal{F} . We further modify Ω' into Ω'' as follows. Letting g be of arity k , we prepare a new variable, say, x and add $g(x, x, \dots, x)$, which essentially works as $e \cdot [1, -1]$ for an appropriate constant $e \neq 0$. A similar argument for Eq.(3) can prove that

$$csp_{\Omega''} = e \cdot csp_{\Omega} + (-1)^{m+1} e \cdot csp_{\Omega} = 2e \cdot csp_{\Omega}. \quad (4)$$

Thus, using the value $csp_{\Omega''}$, we can efficiently compute csp_{Ω} , which equals $\frac{1}{2e}csp_{\Omega''}$. using $csp_{\Omega} = \frac{1}{2}csp_{\Omega'}$ and $csp_{\Omega} = \frac{1}{2e}csp_{\Omega''}$, we can AP-reduce $\#\text{CSP}(\Delta_i, \mathcal{F})$ to $\#\text{CSP}(\mathcal{F})$.

Since the other direction, $\#\text{CSP}(\mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\Delta_i, \mathcal{F})$, is obvious, we finally obtain the desired consequence. \square

In the following subsections, we will concentrate on the proof of Lemma 3.1(2). First, let \mathcal{F} denote any nonempty set of constraints. Obviously, $\#\text{CSP}(\mathcal{F})$ AP-reduces to $\#\text{CSP}(\Delta_i, \mathcal{F})$ for every index $i \in \{0, 1\}$. It therefore suffices to show the other direction (namely, $\#\text{CSP}(\Delta_i, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\mathcal{F})$) for an appropriately chosen index i . Hereafter, we suppose that \mathcal{F} is complement unstable, and we choose a constraint f in \mathcal{F} that is complement unstable. Furthermore, we assume that f has the *smallest arity* k within \mathcal{F} ; that is, there is no complement unstable constraint of arity smaller than k . Our proof of Lemma 3.1(2) proceeds by induction on this index k in Sections 3.2–3.3.

3.2 Basis Case: $k = 1, 2$

Under the assumption described at the very end of Section 3.1, we aim at dealing with the basis case where $k \in \{1, 2\}$. The induction case of $k \geq 3$ will be discussed in Section 3.3.

(1) Assuming $k = 1$, let $f = [x, y]$ with $x, y \in \mathbb{A}$. Note that $x \neq \pm y$. This is because, if $x = \pm y$, then f has the form $x \cdot [1, \pm 1]$ and thus f is complement stable, a contradiction. Hence, it must hold that $|x| \neq |y|$. To appeal to Lemma 2.1(3), it is enough to assert that a certain p-convergence series is effectively T-constructible from f . This assertion comes from the following observation.

Claim 1 *Let x and y be two arbitrary algebraic real numbers with $|x| > |y|$. A p-convergence series $\Lambda = \{[1, |y|^n/|x|^n] \mid n \in \mathbb{N}^+\}$ for Δ_0 is effectively T-constructible from the constraint $[1, |y|/|x|]$. In the case of Δ_1 , a similar statement holds if $|x| < |y|$ (in place of $|x| > |y|$).*

Proof. We set $\lambda = |y|/|x| \geq 0$ and define $g_m = [1, \lambda^m]$ for every number $m \in \mathbb{N}^+$. Since the series $\Lambda = \{g_m \mid m \in \mathbb{N}^+\}$ is a p-convergence series for Δ_0 , the definition of g_m 's leads to the effective T-

constructibility of the constraint Λ from $[1, \lambda]$. □

Since $[1, g_m] \leq_{e\text{-con}} f$ holds for any number $m \in \mathbb{N}^+$, we immediately obtain $\Lambda \leq_{e\text{-con}} f$. Lemma 2.1(3) then leads to Lemma 3.1(2).

(2) Assume that $k = 2$. We intend to prove this case in a slightly more general fashion. Here, we set $f = (x, y, z, w)$ for certain numbers $x, y, z, w \in \mathbb{A}$. We will examine the following two separate cases: $x = \pm w$ and $x \neq \pm w$.

[Case: $x = \pm w$] (a) First, let us assume that $x = w$. Notice that $y \neq z$, because, if $y = z$, f is complement invariant, a contradiction. Since $y \neq z$, we set $g = f^{x_1=*}$, which equals $[x + z, x + y]$. Note that the condition $(x + y)^2 = (x + z)^2$ can be transformed into $(y - z)(2x + y + z) = 0$, which is equivalent to $2x + y + z = 0$ since $y \neq z$. Hence, whenever $2x + y + z \neq 0$, we obtain $|x + y| \neq |x + z|$. If $|x + y| < |x + z|$, then g can approximate $[1, 0]$; otherwise, g approximates $[0, 1]$ by Case (1). Next, we consider the other case where $2x + y + z = 0$. We then start with a new constraint $f' = f^2$ (which equals (x^2, y^2, z^2, w^2)) in place of f . Obviously, it holds that $f' \leq_{e\text{-con}} f$. Let us consider the simple case where $y^2 \neq z^2$. Since f' is not complement stable and $2x^2 + y^2 + z^2 \neq 0$ holds, this case is reduced to the previous case. Finally, let us consider the case where $y^2 = z^2$. Since $y \neq z$, we conclude that $y = -z$. From $2x + y + z = 0$, it instantly follows that $x = 0$. Thus, f equals $(0, y, -y, 0)$, which is complement anti-invariant, a contradiction.

(b) Next, we assume that $x = -w$. First, we claim that $y \neq -z$, because, otherwise, f becomes complement anti-invariant. Let us consider a new constraint $f' = f^2$. If $y^2 \neq z^2$, then f' is not complement stable, and thus we can reduce this case to Case (a). Hence, it suffices to assume that $y^2 = z^2$. This implies $y = z$ and we thus obtain $f = [x, y, -x]$. Now, we define $g = f^{x_1=*} = [x + y, y - x]$. When $(x + y)^2 \neq (y - x)^2$, g can approximate either $[0, 1]$ or $[1, 0]$ because of Case (1). Consider the other case where $(x + y)^2 = (y - x)^2$. This is equivalent to $xy = 0$. If $x = y = 0$, then f is complement invariant. If $x = 0$ and $y \neq 0$, then f is also complement invariant. If $x \neq 0$ and $y = 0$, then f is complement anti-invariant. In any case, we obtain a contradiction.

[Case: $x \neq \pm w$] Clearly, $|x| \neq |w|$ holds. Let us define $g = f^{x_1=x_2}$, which equals $[x, w]$. If $|x| < |w|$, then g equals $z \cdot h$, where $h = [\frac{x}{w}, 1]$. Since $|\frac{x}{w}| < 1$, h can approximate $[0, 1]$ by Case (1). Otherwise, g equals $x \cdot h'$, where $h' = [1, \frac{w}{x}]$. By Case (1), this h' can approximate $[1, 0]$. Therefore, we obtain $\#CSP(\Delta_i, \mathcal{F}) \equiv_{AP} \#CSP(\mathcal{F})$ for every index $i \in \{0, 1\}$.

3.3 Induction Case: $k \geq 3$

We will examine the remaining case of $k \geq 3$. As the next lemma indicates, from a given complement unstable constraint f of arity k , we can effectively T-construct another complement unstable constraint g of arity less than k .

Lemma 3.2 *Let $k \geq 3$ and let f be any k -ary constraint. If f is complement unstable, then there exists another constraint g of arity less than k for which $g \leq_{e\text{-con}} f$ and g is also complement unstable.*

Assuming that Lemma 3.2 is true, we take a complement unstable constraint g of arity $< k$. Instead of working on the originally given set \mathcal{F} , let us concentrate on another set $\mathcal{F}' = \mathcal{F} \cup \{g\}$. Since g has arity less than k , the induction hypothesis asserts that a certain constant unary constraint Δ_i ($i \in \{0, 1\}$) satisfies $\#CSP(\Delta_i, \mathcal{F}') \equiv_{AP} \#CSP(\mathcal{F}')$. It follows from both $g \leq_{e\text{-con}} f$ and $f \in \mathcal{F}$ that $\#CSP(\Delta_i, \mathcal{F}) \leq_{AP} \#CSP(g, \mathcal{F}) \leq_{AP} \#CSP(f, \mathcal{F}) \leq_{AP} \#CSP(\mathcal{F})$, implying $\#CSP(\Delta_i, \mathcal{F}) \leq_{AP} \#CSP(\mathcal{F})$ by Lemma 2.1. This completes the induction case. Therefore, the remaining task of ours is to give the proof of Lemma 3.2.

Proof of Lemma 3.2. Let $f = (z_1, z_2, \dots, z_{2k})$. Since f is complement unstable, there exists an index $\ell \in [2k]$ satisfying $z_\ell \neq 0$. For each pair of indices $i, j \in [k]$ with $i < j$, we define a new constraint $g^{(i,j)}$ to be $f^{x_i=x_j}$ and then we set $\mathcal{G} = \{g^{(i,j)} \mid 1 \leq i < j \leq k\}$. Note that each constraint $g^{(i,j)}$ is effectively T-constructible from f . Therefore, if \mathcal{G} contains a complement unstable constraint, say, g , then g has arity $k - 1$ and $g \leq_{e\text{-con}} f$ holds. The lemma instantly follows. In what follows, we will examine the case where every constraint in \mathcal{G} is complement stable.

Let us begin with a simple observation.

Claim 2 *Let $k \geq 3$. For any index $j \in [2^k]$, there exist a constraint g in \mathcal{G} and $k - 1$ bits a_1, a_2, \dots, a_{k-1} satisfying $z_j = g(a_1, a_2, \dots, a_{k-1})$.*

Proof. Since z_j is an output value of f , there exists an input tuple $(a'_1, a'_2, \dots, a'_k) \in \{0, 1\}^k$ such that $z_j = f(a'_1, a'_2, \dots, a'_k)$. Since $k \geq 3$, there are two indices i, j with $i < j$ satisfying $a'_i = a'_j$. For this special pair, it follows that $g^{(i,j)}(a'_1, \dots, a'_{i-1}, a'_{i+1}, \dots, a'_k) = f(a'_1, \dots, a'_{i-1}, a'_j, a'_{i+1}, \dots, a'_k) = z_j$. It therefore suffices to set the desired constraint g to be $g^{(i,j)}$ and set $(a_1, a_2, \dots, a_{k-1})$ to be $(a'_1, \dots, a'_{i-1}, a'_{i+1}, \dots, a'_k)$. \square

Let us return to the proof of Lemma 3.2. Since f is complement unstable, either of the following two cases must occur. (1) There exists an index $i \in [2^{k-1}]$ satisfying $|z_i| \neq |z_{2^k-i+1}|$. (2) It holds that $|z_i| = |z_{2^k-i+1}|$ for every index $i \in [2^{k-1}]$, but there are two distinct indices $i_0, j_0 \in [2^{k-1}]$ for which $z_{i_0} = z_{2^k-i_0+1} \neq 0$ and $z_{j_0} = -z_{2^k-j_0+1} \neq 0$.

(1) In the first case, let us choose an index $i \in [2^{k-1}]$ satisfying $|z_i| \neq |z_{2^k-i+1}|$. Claim 2 ensures the existence of a constraint g in \mathcal{G} such that $z_i = g(a_1, a_2, \dots, a_{k-1})$ for appropriately chosen $k-1$ bits a_1, a_2, \dots, a_{k-1} . This implies that $z_{2^k-i+1} = g(a_1 \oplus 1, a_2 \oplus 1, \dots, a_{k-1} \oplus 1)$. By the choice of i , g cannot be complement stable. However, this is a contradiction against our assumption that \mathcal{G} is complement stable.

(2) In the second case, let us take any two indices $i_0, j_0 \in [2^{k-1}]$ satisfying that $z_{i_0} = z_{2^k-i_0+1} \neq 0$ and $z_{j_0} = -z_{2^k-j_0+1} \neq 0$. We will examine two possible cases separately.

(i) Assume that a certain constraint $g \in \mathcal{G}$ satisfies both $z_{i_0} = g(a_1, a_2, \dots, a_{k-1})$ and $z_{j_0} = g(b_1, b_2, \dots, b_{k-1})$ for certain $2(k-1)$ bits $a_1, a_2, \dots, a_{k-1}, b_1, b_2, \dots, b_{k-1}$. From the properties of z_{i_0} and z_{j_0} , it follows that g is complement unstable, and this fact clearly leads to a contradiction.

(ii) Finally, assume that Case (i) does not hold. This case is much more involved than Case (i). By our assumption, $|z_i| = |z_{2^k-i+1}|$ holds for all indices $i \in [2^{k-1}]$. To make the following argument simple, we will introduce several notations. First, we denote by H' the set of all index pairs (i, j) in $[2^k] \times [2^k]$ such that both $z_i = g(a_1, a_2, \dots, a_{k-1})$ and $z_j = g(b_1, b_2, \dots, b_{k-1})$ hold for a certain constraint g in \mathcal{G} and certain $2(k-1)$ bits $a_1, a_2, \dots, a_{k-1}, b_1, b_2, \dots, b_{k-1}$. Associated with H' , we set $H = [2^k] \times [2^k] - H'$ and define $\hat{H} = \{i \in [2^k] \mid \exists j[(i, j) \in H]\}$. This set \hat{H} can be expressed as the union $\hat{H}_0 \cup \hat{H}_+ \cup \hat{H}_-$, where $\hat{H}_0 = \{i \in \hat{H} \mid z_i = z_{2^k-i+1} = 0\}$, $\hat{H}_+ = \{i \in \hat{H} \mid z_i = z_{2^k-i+1} \neq 0\}$, and $\hat{H}_- = \{i \in \hat{H} \mid z_i = -z_{2^k-i+1} \neq 0\}$. Concerning \hat{H}_+ and \hat{H}_- , the following useful properties hold.

Claim 3 Let $i, j \in [2^k]$ be any two indices with $i \leq 2^{k-1}$ and assume that $(i, j) \in H'$.

1. If $j \in \hat{H}_+$ then $z_i = z_{2^k-i+1}$ holds.
2. If $j \in \hat{H}_-$ then $z_i = -z_{2^k-i+1}$ holds.

Proof. If $z_i = 0$, then $z_{2^k-i+1} = 0$ follows because of $|z_i| = |z_{2^k-i+1}|$. Since $z_i = \pm z_{2^k-i+1}$, the claim is trivially true. Henceforth, we consider the case where $z_i \neq 0$. Since $(i, j) \in H'$, there exist a constraint $g \in \mathcal{G}$ and bits $a_1, a_2, \dots, a_{k-1}, b_1, b_2, \dots, b_{k-1}$ for which $z_i = g(a_1, a_2, \dots, a_{k-1})$ and $z_j = g(b_1, b_2, \dots, b_{k-1})$. Note that either $z_i = z_{2^k-i+1}$ or $z_i = -z_{2^k-i+1}$ holds since g is complement stable by our assumption. In the case where $j \in \hat{H}_+$, since $z_j = z_{2^k-j+1} \neq 0$ holds, g must be complement invariant. Thus, for the index i , we obtain $z_i = z_{2^k-i+1}$. By a similar argument, when $j \in \hat{H}_-$, g must be complement anti-invariant and thus $z_i = -z_{2^k-i+1}$ holds. \square

Here, we claim that \hat{H}_+ and \hat{H}_- are both nonempty. To see this claim, recall that the pair (i_0, j_0) satisfies that $(i_0, j_0) \notin H'$, and thus they belong to \hat{H} . More specifically, it holds that $i_0 \in \hat{H}_+$ and $j_0 \in \hat{H}_-$ since $z_{i_0} = z_{2^k-i_0+1}$ and $z_{j_0} = -z_{2^k-j_0+1}$. By symmetry, we also conclude that $2^k - i_0 + 1 \in \hat{H}_+$ and $2^k - j_0 + 1 \in \hat{H}_-$. Moreover, in Claim 4, we will present another useful property of \hat{H} .

Claim 4 For any index $i \in [2^k]$, if $i \notin \hat{H}$, then $z_i = 0$ holds.

Proof. Let i be any index in $[2^k] - \hat{H}$. Toward a contradiction, we assume that $z_i \neq 0$. Since \hat{H}_+ and \hat{H}_- are nonempty, let us take two indices $j_1 \in \hat{H}_+$ and $j_2 \in \hat{H}_-$ and consider two pairs (i, j_1) and (i, j_2) . Regarding to the first pair (i, j_1) , if $(i, j_1) \notin H'$ holds, then (i, j_1) must be in H , and thus i belongs to \hat{H} . Since this is clearly a contradiction, $(i, j_1) \in H'$ follows. Similarly, we can obtain $(i, j_2) \in H'$ for the second pair (i, j_2) . Claim 3 then implies $z_i = z_{2^k-i+1}$ as well as $z_i = -z_{2^k-i+1}$. From these equations, $z_i = 0$ follows, a contradiction. Therefore, the claim is true. \square

The rest of the proof proceeds by examining three cases, depending on the value of k .

(a) Let us consider the base case of $k = 3$ with $f = (z_1, z_2, \dots, z_8)$. By a straightforward calculation, \hat{H} equals $\{2, 3, 4, 5, 6, 7\}$. Claim 4 yields the equality $z_1 = z_8 = 0$. Now, we assume that $\hat{H}_0 \neq \emptyset$. In the case where $\hat{H}_0 = \{4, 5\}$, we define $h_2 = f^{x_2=*}$, which equals (z_3, z_2, z_7, z_6) . If h_2 is complement stable, then it

must hold that either $z_i = z_{9-i}$ for all $i \in [4]$ or $z_i = -z_{9-i}$ for all $i \in [4]$. This implies that f is complement stable, a contradiction. The other cases ($\hat{H}_0 = \{2, 7\}$ and $\hat{H}_0 = \{3, 6\}$) are similar.

Now, we assume that $\hat{H}_0 = \emptyset$. Recall that $|\hat{H}_+| > 0$ and $|\hat{H}_-| > 0$. Notice that $|\hat{H}_+| \neq |\hat{H}_-|$. Let us consider the case where $|\hat{H}_+| > |\hat{H}_-|$. If $\hat{H}_+ = \{2, 4, 5, 7\}$ and $\hat{H}_- = \{3, 6\}$, then we define $h_2 = f^{x_2=*}$, which equals $(z_3, z_2 + z_4, z_5 + z_7, z_6)$. Since $3 \in \hat{H}_-$, $z_3 = -z_6$ holds; moreover, since $2, 4 \in \hat{H}_+$, it follows that $z_2 + z_4 = z_5 + z_7$. We then conclude that h_2 is not complement stable. Similarly, if $\hat{H}_+ = \{2, 3, 6, 7\}$ and $\hat{H}_- = \{4, 5\}$ (resp., $\hat{H}_+ = \{3, 4, 5, 6\}$ and $\hat{H}_- = \{2, 7\}$), then consider $h_1 = (z_5, z_2 + z_6, z_3 + z_7, z_4)$ (resp., $h_3 = (z_2, z_3 + z_4, z_5 + z_6, z_7)$). The other case where $|\hat{H}_-| > |\hat{H}_+|$ is similar.

(b) Consider the case where $k = 4$. It is not difficult to show that $\hat{H} = \{4, 6, 7, 10, 11, 13\}$. Next, we consider $h = f^{x_1=*}$. This $h = (w_1, w_2, \dots, w_8)$ contains entries $w_i = z_i + z_{8+i}$ for all $i \in [8]$. In particular, $w_1 = z_1 + z_9$, $w_2 = z_2 + z_{10}$, $w_3 = z_3 + z_{11}$, $w_4 = z_4 + z_{12}$, $w_5 = z_5 + z_{13}$, $w_6 = z_6 + z_{14}$, $w_7 = z_7 + z_{15}$, and $w_8 = z_8 + z_{16}$. By Claim 4, it follows that $h = (0, z_{10}, z_{11}, z_4, z_{13}, z_6, z_7, 0)$. If h is complement stable, either $z_i = z_{16-i+1}$ for all $i \in \{4, 6, 7\}$ or $z_i = -z_{16-i+1}$ for all $i \in \{4, 6, 7\}$. This is impossible because $i_0, j_0 \in \hat{H}$.

(c) Assume that $k \geq 5$. We show that $H = \emptyset$. Assume that $(i, j) \in H$ and i and j respectively correspond to k bits $a = a_1 a_2 \dots a_k$ and $b = b_1 b_2 \dots b_k$. Since a and b match at most one location for each bit 0 and 1, we obtain $b_i = \bar{a}_i$ for all but two positions of i 's. Let P be the set of such locations. Since $k \geq 5$, for at least two locations $i, j \in P$, b_i and b_j coincide. This fact implies that $(i, j) \in H'$, a contradiction.

This completes the proof of Lemma 3.2 and thus finishes the proof of Theorem 1.1. \square

Throughout the proof of Theorem 1.1, we have required the use of algebraic real numbers only in the proof of Claim 1. It is not known so far that the theorem is still true for arbitrary real numbers.

4 AP-Reductions without Auxiliary Unary Constraints

As a direct application of Theorem 1.1, we wish to prove our second main theorem—Theorem 1.3—presented in Section 1. To clarify the meaning of this theorem, we need to introduce the following sets of constraints. Recall that all constraints dealt with in this paper are assumed to output only *algebraic real values*.

1. Let \mathcal{DG} denote the set of all constraints f that are expressed by products of unary functions. A constraint in \mathcal{DG} is called *degenerate*. When f is symmetric, f must have one of the following three forms: $[x, 0, \dots, 0]$, $[0, \dots, 0, x]$, and $y \cdot [1, z, z^2, \dots, z^k]$ with $yz \neq 0$. By restricting \mathcal{DG} , we define $\mathcal{DG}^{(-)}$ as the set of constraints of the forms $[x, 0, \dots, 0]$, $[0, \dots, 0, x]$, $y \cdot [1, 1, \dots, 1]$, and $y \cdot [1, -1, 1, \dots, -1 \text{ or } 1]$, where $y \neq 0$. Naturally, both Δ_0 and Δ_1 belong to $\mathcal{DG}^{(-)}$.
2. The notation \mathcal{ED}_1 indicates the set of the following constraints: $[x, \pm x]$, $[x, 0, \dots, 0, \pm x]$ of arity ≥ 2 , and $[0, x, 0]$ with $x \neq 0$. As a natural extension of \mathcal{ED}_1 , let $\mathcal{ED}_1^{(+)}$ be composed of constraints $[x, y]$, $[x, 0, \dots, 0, y]$ of arity ≥ 2 , and $[0, x, 0]$ with $x, y \neq 0$. Notice that $[x, y]$ also belongs to \mathcal{DG} .
3. Let \mathcal{AZ} be made up of all constraints of arity at least 3 having the forms $[0, x, 0, x, \dots, 0 \text{ or } x]$ and $[x, 0, x, 0, \dots, x \text{ or } 0]$ with $x \neq 0$. The term “ \mathcal{AZ} ” indicates “alternating zeros.”
4. Let \mathcal{OR} denote the set of all constraints of the form $[0, x, y]$ with $x, y > 0$. For instance, a special constraint $OR = [0, 1, 1]$ belongs to \mathcal{OR} .
5. Let \mathcal{NAND} consist of all constraints of the form $[x, y, 0]$ with $x, y > 0$. A Boolean constraint $NAND = [1, 1, 0]$ is in \mathcal{NAND} .
6. Let \mathcal{B} be comprised of all constraints of the form $[x, y, z]$ with $x, y, z \neq 0$ and $xz \neq y^2$.

In the presence of \mathcal{U} , it holds that $\#\text{SAT} \leq_{\text{AP}} \#\text{CSP}(g, \mathcal{U})$ for every constraint g in $\mathcal{OR} \cup \mathcal{B}$ [11]. Moreover, following an argument for the proof of [11, Proposition 6.2] that asserts the AP-equivalence $\#\text{CSP}(OR, \mathcal{U}) \equiv_{\text{AP}} \#\text{CSP}(NAND, \mathcal{U})$, it is possible to prove that \mathcal{NAND} and \mathcal{OR} are similar in approximation complexity *without* the presence of \mathcal{U} .

Lemma 4.1 *For any $f \in \mathcal{NAND}$ (resp., \mathcal{OR}), there exists a constraint $g \in \mathcal{OR}$ (resp., \mathcal{NAND}) such that $\#\text{CSP}(g) \leq_{\text{AP}} \#\text{CSP}(f)$.*

The first part of Theorem 1.3 concerns the tractability of $\#\text{CSP}(\mathcal{F})$ when either $\mathcal{F} \subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$ or $\mathcal{F} \subseteq \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$ holds. For such a counting problem $\#\text{CSP}(\mathcal{F})$, we will demonstrate how to solve $\#\text{CSP}(\mathcal{F})$ in polynomial time.

Proposition 4.2 *Let \mathcal{F} be any set of symmetric real-valued constraints. If either $\mathcal{F} \subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$ or $\mathcal{F} \subseteq \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$ holds, then $\#\text{CSP}(\mathcal{F})$ belongs to $\text{FP}_{\mathbb{A}}$.*

Before giving the proof of Proposition 4.2, we will present a core portion of the proof as a separate lemma that targets a *single* constraint rather than a constraint set \mathcal{F} .

Lemma 4.3 *If f is in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, then $\#\text{CSP}(f)$ is in $\text{FP}_{\mathbb{A}}$.*

Proof. Given any input constraint frame Ω , we explain how to modify the input graph so that we can compute csp_{Ω} easily. Let $X = \{x_1, x_2, \dots, x_n\}$ be the set of all Boolean variables appearing in Ω .

(1) If f is in \mathcal{DG} , then we decompose f into a product of several unary constraints and replace this f in the graph by those unary constraints. Since all constraints become unary in the input graph, the value csp_{Ω} can be easily computed.

(2) If f belongs to $\mathcal{ED}_1^{(+)}$, then (assuming $f(x_1, \dots, x_k)$ and $f = [x, 0, \dots, 0, y]$ with $xy \neq 0$) we merge all nodes labeled x_1, \dots, x_k into a single node with a new variable label and replace f with $[x, y]$. Note that $[0, x, 0]$ equals $x \cdot [0, 1, 0]$, which is $x \cdot \text{XOR}$. Since x is a constant, it suffices to use XOR or $[x, y]$. Since the obtained graph therefore contains only XOR or $[x, y]$, we partition the graph into connected components. For each connected component, say, Ω' , $\text{csp}_{\Omega'}$ can be easily computed. Since the total value csp_{Ω} is thus the product of all such $\text{csp}_{\Omega'}$'s, it is easy to compute csp_{Ω} .

(3) Assume that $f \in \mathcal{AZ}$. Note that f has the form $x \cdot [1, 0, 1, 0, \dots, 1 \text{ or } 0]$ or $x \cdot [0, 1, 0, 1, \dots, 0 \text{ or } 1]$ with $x \neq 0$. Since the constant x can be computed globally, it is possible to set $x = 1$ in the following argument. When f has arity k and is of the form $[0, 1, 0, 1, \dots, 0 \text{ or } 1]$, it holds that $f(x_1, \dots, x_k) = \sum_{i=1}^k x_i \bmod 2$. Hence, csp_{Ω} is the sum, over all possible variable assignments, of a product of the terms, each of which has the form $\sum_{i \in I_j} x_i \bmod 2$, where I_j is a subset of X . This product can be written as a sum of monomials of the form $x_{i_1} x_{i_2} \cdots x_{i_j}$ on $GF(2)$. Let I denote the set of index series (i_1, i_2, \dots, i_j) corresponding to those monomials. Hence, csp_{Ω} is the sum, over all index series $(i_1, i_2, \dots, i_j) \in I$, of $\sum_{\sigma} \sigma(x_{i_1}) \sigma(x_{i_2}) \cdots \sigma(x_{i_j})$, where σ ranges all possible variable assignments. Clearly, this value can be computed in polynomial time. Therefore, it holds that $\#\text{CSP}(f) \in \text{FP}_{\mathbb{A}}$. On the contrary, when f has the form $[1, 0, 1, 0, \dots, 1 \text{ or } 0]$, it follows that $f(x_1, \dots, x_k) = \sum_{i=1}^k x_i + 1 \bmod 2$. The rest of the argument is similar to the previous case. \square

We will prove Proposition 4.2 based on the proof of Lemma 4.3.

Proof of Proposition 4.2. To simplify our argument below, we will refer to Cases (1)–(3) of the proof of Lemma 4.3.

First, let us consider the case where $\mathcal{F} \subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$. As shown in Cases (1)–(2), it is possible to assume that all constraints are unary constraints or XOR . It is therefore easy to compute csp_{Ω} in a way similar to Cases (1)–(2).

Next, consider the case where $\mathcal{F} \subseteq \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$. We claim that it is sufficient to assume that \mathcal{F} contains only the following constraints: $[0, 1, 0, 1, \dots, 0 \text{ or } 1]$, $[1, 0, 1, 0, \dots, 1 \text{ or } 0]$, $[0, 1, 0]$, $[1, -1]$, $[0, 1]$, and $[1, 0]$ without any global constant. This is shown as follows. If f is in $\mathcal{DG}^{(-)} \cup \mathcal{ED}_1$, we use a modification of an input graph described as in Cases (1)–(2) to make f satisfy the form described above. If f is in \mathcal{AZ} , then we leave out a global constant as in Case (3).

Similarly to Case (3), we replace $[0, 1, 0, 1, \dots, 0 \text{ or } 1]$ and $[1, 0, 1, 0, \dots, 1 \text{ or } 0]$ by $\sum_{i \in I_j} x_i \bmod 2$ and $\sum_{i \in I_j} x_i + 1 \bmod 2$, respectively. Moreover, we replace $[0, 1, 0]$ by $x_i + x_j \bmod 2$, $[1, -1]$ by $(-1)^{x_i}$, $[1, 0]$ by x_i , and $[0, 1]$ by $1 - x_i$, where x_i and x_j are corresponding variables. The product of all those constraints can be expressed as a sum of monomials of the form $(-1)^{x_{i_1} + \cdots + x_{i_j}} x_{i_1} x_{i_2} \cdots x_{i_j}$. An idea similar to Case (3) then helps compute the value csp_{Ω} . \square

Now, we come to the point of proving the second part of Theorem 1.3. Let us first analyze the approximation complexity of $\#\text{CSP}(f)$ for an arbitrary symmetric constraint f that are not in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. Note that, when f is in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, $\#\text{CSP}(f)$ belongs to $\text{FP}_{\mathbb{A}}$ by Proposition 4.2. A key claim required for the proof of Theorem 1.3 is the following assertion.

Lemma 4.4 *Let f be any symmetric real-valued constraint of arity at least 2. If $f \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, then there exists a constraint $g \in \mathcal{OR} \cup \mathcal{B}$ such that $\#\text{CSP}(g) \leq_{\text{AP}} \#\text{CSP}(f)$.*

Theorem 1.3 then follows directly by combining Theorem 1.1 and Proposition 4.2 with an application of Lemma 4.4.

Proof of Theorem 1.3. Since Proposition 4.2 has already proven the first part of Theorem 1.3, we intend to prove the last part of the theorem. Toward this goal, we assume that $\mathcal{F} \not\subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$ and $\mathcal{F} \not\subseteq \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$. Notice that \mathcal{F} should contain a certain constraint whose entries are not all zero.

If there exists a constraint f in \mathcal{F} satisfying $f \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, then we apply Lemma 4.4 to obtain the theorem. Hereafter, we assume that $\mathcal{F} \subseteq \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. From this assumption, we can choose two constraints $f_1, f_2 \in \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$ in \mathcal{F} for which $f_1 \notin \mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}$ and $f_2 \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)}$. Note that $f_1 \in \mathcal{DG} - \mathcal{ED}_1$ and $f_2 \in \mathcal{AZ}$. Hereafter, we will prove the following claim.

Claim 5 *There exists a constraint g in $\mathcal{OR} \cup \mathcal{B}$ such that $\#CSP(g) \leq_{AP} \#CSP(f_1, f_2)$.*

The proof of this claim proceeds as follows. Since $f_2 \in \mathcal{AZ}$, there are two cases, $f_2 = [0, 1, 0, 1, \dots, 0 \text{ or } 1]$ and $f_2 = [1, 0, 1, 0, \dots, 1 \text{ or } 0]$, to examine; however, since either Δ_0 or Δ_1 is available by Theorem 1.1, f_2 can be reduced to either $[1, 0, 1, 0]$ or $[0, 1, 0, 1]$. Here, let us consider only the case where $f_2 = [1, 0, 1, 0]$ because the other case is similarly handled.

(1) Assume that $f_1 = [x, y]$ with $xy \neq 0$. From $f_1 \notin \mathcal{ED}_1$, it follows that $x \neq \pm y$. Define $g(x_1, x_2, x_3) = f_1(x_1)f_2(x_1, x_2, x_3)$ and $h(x_1, x_2, x_3) = g(x_1, x_2, x_3)g(x_2, x_3, x_1)g(x_3, x_1, x_2)$. A simple calculation shows that h equals $[x^2, 0, y^2, 0]$. Since $x^2 \neq y^2$, we conclude that $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. We apply Lemma 4.4 and then obtain the desired consequence.

(2) Assume that $f_1 = y \cdot [1, z, z^2, \dots, z^k]$ with $y \neq 0$ and $z \neq \pm 1$. Using either Δ_0 or Δ_1 , we can assume that f_1 is of the form $[1, z]$. Thus, this case is reduced to (1). \square

Since $\mathcal{U} \subseteq \mathcal{DG}$, the problem $\#CSP(\mathcal{DG} \cup \mathcal{ED}_1^{(+)}, \mathcal{U})$ coincides with $\#CSP(\mathcal{DG} \cup \mathcal{ED}_1^{(+)})$ and, by Proposition 4.2, it is solvable in polynomial time. On the contrary, $\#CSP(\mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}, \mathcal{U})$ turns out to possess much higher complexity than $\#CSP(\mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ})$. Notice that $\#CSP(\mathcal{DG}^{(-)} \cup \mathcal{ED}_1 \cup \mathcal{AZ}, \mathcal{U})$ is AP-equivalent to $\#CSP(\mathcal{DG} \cup \mathcal{ED}_1 \cup \mathcal{AZ}, \mathcal{U})$. Claim 5 actually shows that $\#CSP(g) \leq_{AP} \#CSP(\mathcal{DG} \cup \mathcal{AZ})$; thus, it holds that $\#CSP(g, \mathcal{U}) \leq_{AP} \#CSP(\mathcal{DG} \cup \mathcal{AZ}, \mathcal{U})$.

To finish our entire argument, we still need to prove Lemma 4.4.

Proof of Lemma 4.4. Let f be any symmetric real-valued constraint of arity $k \geq 2$. Throughout this proof, we assume that $f \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. Our proof proceeds by induction on k .

Case of $k = 2$. Let f be any binary constraint not in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)}$. There are three major cases to consider separately, depending on the number of zeros in the output values of f .

(B1) Consider the case where there are two zeros in the entries of f . Obviously, f must have one of the following forms: $[x, 0, 0] \in \mathcal{DG}$, $[0, 0, x] \in \mathcal{DG}$, and $[0, x, 0] \in \mathcal{ED}_1$, yielding a contradiction against the assumption on f .

(B2) Consider the case where there is exactly one zero in f . Here, f has one of the following forms: $[0, x, y]$, $[x, 0, y]$, and $[x, y, 0]$, where $xy \neq 0$. For the first and the last forms, f^2 respectively belongs to \mathcal{OR} and \mathcal{NAND} . If $f^2 \in \mathcal{NAND}$, then we apply Lemma 4.1 and obtain a certain g in \mathcal{OR} such that $\#CSP(g) \leq_{AP} \#CSP(f^2)$. If $f^2 \in \mathcal{OR}$, then we choose f^2 as the desired g . Since $[x, 0, y]$ is in $\mathcal{ED}_1^{(+)}$, the lemma holds.

(B3) Finally, consider the case where there is no entry of zero in f . When $xz \neq y^2$, f belongs to \mathcal{B} . We set g to be f itself. If $xz = y^2$, then f has the form $[x, y, y^2/x]$ and thus it is in \mathcal{DG} . In both cases, the lemma is true.

Case of $k = 3$. Now, we assume that f has arity 3. For convenience, notations x, y, z, w that will appear as real values are assumed to be non-zero.

(T1) Consider the case where f has exactly three zeros; that is, f is one of the following four forms: $[x, 0, 0, 0]$, $[0, x, 0, 0]$, $[0, 0, x, 0]$, and $[0, 0, 0, x]$. If $f \in \{[x, 0, 0, 0], [0, 0, 0, x]\}$, then f falls into \mathcal{DG} , a contradiction. Now, assume that $f = [0, x, 0, 0]$. If $f = [0, x, 0, 0]$, then we define $h = f^{x_1=*}$, which equals $x \cdot [1, 1, 0]$. This constraint h belongs to \mathcal{NAND} . By Lemma 4.1, there exists a constraint g in \mathcal{OR} such

that $\#CSP(g) \leq_{AP} \#CSP(h)$. Therefore, it follows that $\#CSP(g) \leq_{AP} \#CSP(f)$. The case $f = [0, 0, x, 0]$ is treated more directly using \mathcal{OR} .

(T2) Let us consider the case where f has exactly two zeros: $f = [x, 0, 0, y], [x, y, 0, 0], [0, 0, x, y], [x, 0, y, 0], [0, x, 0, y], [0, x, y, 0]$. Theorem 1.1 guarantees that either Δ_0 or Δ_1 is available to use. Note that $[x, 0, 0, y]$ does not belong to \mathcal{ED}_1 .

(a) If $f = [x, y, 0, 0]$, then we define $g = f^{x_1=x_2=x_3}$, which yields $x \cdot \Delta_0$. Since $x \neq 0$, we can freely use Δ_0 . We can effectively T-construct $h = [x, y, 0]$ from $\{f, \Delta_0\}$. Therefore, h^2 is also effectively T-constructible. It is obvious that $h^2 = [x^2, y^2, 0]$ is in \mathcal{NAND} . The case $f = [0, 0, x, y]$ is handled similarly using \mathcal{OR} instead of \mathcal{NAND} .

(b) When $f = [0, x, y, 0]$, since either Δ_0 or Δ_1 is available from f , we can effectively T-construct either $[0, x, y]$ or $[x, y, 0]$, which is reduced to Case (B2).

(c) Finally, assume that $f = [x, 0, y, 0]$. In the case where $x = y$, we obtain $f = x \cdot [1, 0, 1, 0]$, which belongs to \mathcal{AZ} . If $x \neq y$, then we consider $g = f^{x_1=*}$, which equals $[x, y, y]$. Note that $x = y$ iff $xy = y^2$. Since $x \neq y$, g must belong to \mathcal{B} . The case $f = [0, x, 0, y]$ is similar.

(T3) Consider the case where f has exactly one zero: $f = [x, y, z, 0], [0, x, y, z], [x, y, 0, z], [x, 0, y, z]$.

(a) If f is of the form $[x, y, z, 0]$, then we define $g = f^{x_1=x_2}$, which equals $(x, y, z, 0)$. We then define $h(x_1, x_2) = g(x_1, x_2)^2 g(x_2, x_1)^2$, implying $h = [x^4, y^2 z^2, 0]$. This h belongs to \mathcal{NAND} . By duality, we effectively T-construct $[0, xy, z^2]$ from $[0, x, y, z]$.

(b) If $f = [x, 0, y, z]$, then define $g(x_1, x_2) = \sum_{x_3, x_4 \in \{0, 1\}} f(x_1, x_3, x_4) f(x_2, x_3, x_4)$. This implies that $g = [A, B, C]$, where $A = x^2 + y^2$, $B = yz$, $C = 2y^2 + z^2$. We want to claim that $AC \neq B^2$. To show this inequality, assume that $AC = B^2$; that is, $(x^2 + y^2)(2y^2 + z^2) = y^2 z^2$. This implies $2y^4 + 2x^2 y^2 + x^2 z^2 = 0$. Since $z^2 = -\frac{2y^2(x^2 + y^2)}{x^2}$, it follows that $z^2 < 0$, a contradiction. Hence, the claim holds and g belongs to \mathcal{B} . By duality, we similarly handle the case where $f = [x, y, 0, z]$.

(T4) Let us consider the case where f has no zero; namely, $f = [x, y, z, w]$ with $xyzw \neq 0$. Theorem 1.1 guarantees the use of either Δ_0 or Δ_1 .

(a) If $xz \neq y^2$ and $yw \neq z^2$, then we obtain either $[x, y, z]$ or $[y, z, w]$ using Δ_0 or Δ_1 , respectively. This case can be reduced to Case (B3).

(b) Assume that $xz \neq y^2$ and $yw = z^2$. For simplicity, without loss of generality, we assume that $x = 1$. Define $h(x_1, x_2) = \sum_{x_3, x_4 \in \{0, 1\}} f(x_1, x_3, x_4) f(x_2, x_3, x_4)$, which equals $[1 + 2y^2 + z^2, y + 2yz + zw, y^2 + 2z^2 + w^2]$. There are two cases to consider.

(i) We consider the first case where $\frac{z^4}{y^2} \neq y^2 z^2$. Obviously, h belongs to \mathcal{B} . Since $h \leq_{c-con} f$, the lemma follows immediately.

(ii) The second case is that $\frac{z^4}{y^2} = y^2 z^2$. This equality yields $z^2 = y^4$. Since $z \neq y^2$, we obtain $z = -y$. Note that h is of the form $[A, B, C]$, where $A = \frac{1}{z^2}(y^2 + z^2)$, $B = \frac{1}{yz}(z^4 + 2y^2 z^2 - y^4)$, and $C = \frac{1}{y^2}(y^2 + z^2)$. If $AC = B^2$, then we obtain $z^2 = y^2(y^2 - 1)$. From the equality $z = -y$, we obtain $y^4 = y^2(y^2 - 1)$, which yields $y = 0$, a contradiction. Hence, we conclude that $AC \neq B^2$. This implies that h is in \mathcal{B} .

Case of $k \geq 4$. There are four fundamental cases to study. For convenience, let $u = f(0^k)$ and $w = f(1^k)$.

[Case: $u = 0$ and $w \neq 0$] Since the other case where $u \neq 0$ and $w = 0$ is symmetric, we omit this case. Now, we begin with defining $h' = f^{x_1=x_2=\dots=x_k}$; that is, $h' = [0, w]$. Since the constraint h' can approximate $[0, 1]$, we can freely use Δ_1 . We set g as $f^{x_1=1}$ (equivalently, $\sum_{x_1 \in \{0, 1\}} f(x_1, \dots, x_k) \Delta_1(x_1)$). Note that, if $g \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, then we obtain the desired consequence. In what follows, we assume that $g \in \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$.

(a) In the case where $g \in \mathcal{DG}$, g cannot be $[x, 0, 0, \dots, 0]$ because $w \neq 0$. If $g = [0, 0, \dots, 0, x]$, then f equals $[0, 0, 0, \dots, 0, x]$, which belongs to \mathcal{DG} , a contradiction. The remaining case is that $g = x \cdot [1, y, y^2, \dots, y^k]$ with $xy \neq 0$. If $y \neq -1$, then we define $h = f^{x_1=*}$, which is $x \cdot [1, y + 1, y(y + 1), \dots, y^{k-1}(y + 1)]$. Since $y(y + 1) \neq (y + 1)^2$, h is not in \mathcal{DG} . We then apply the induction hypothesis to obtain the desired result. On the contrary, when $y = -1$, we consider $f^2 = x^2 \cdot [0, 1, 1, \dots, 1]$. This case can be reduced to the previous case of $y \neq -1$.

(b) Let us consider the case where $g \in \mathcal{ED}_1^{(+)}$. Note that g is not of the form $[0, x, 0]$ because g 's arity cannot be 2. If $g = [x, 0, \dots, 0, w]$, then f must be $[0, x, 0, \dots, 0, w]$. Let $h = f^{x_1=*}$, implying $h = [x, x, 0, \dots, 0, w]$. Since $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, the induction hypothesis leads to the desired consequence.

(c) The final case to consider is that $g \in \mathcal{AZ}$. The cases $g = [x, 0, x, 0, \dots, x, 0]$ and $g = [0, x, 0, x, \dots, x, 0]$ do not occur because $w \neq 0$. If g has the form $[x, 0, x, 0, \dots, x]$, then f must be of the form $[0, x, 0, x, 0, \dots, x]$. The constraint f thus belongs to \mathcal{AZ} , a contradiction. Moreover, if $g = [0, x, 0, x, \dots, 0, x]$, then f equals

$[0, 0, x, 0, x, \dots, x, 0]$. Now, we define $h = f^{x_1=*}$, which equals $x \cdot [0, 1, 1, \dots, 1]$. Since $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, we then apply the induction hypothesis to obtain the desired result.

[Case: $uw \neq 0$]

(i) Assume that $|u| = |w|$. If $u = -w$, then we consider $f' = f^2$. This constraint f' satisfies $f'(0, \dots, 0) = f'(1, \dots, 1)$, and thus it falls into the case of $u = w$. Thus, it suffices to discuss the case $u = w$. By Theorem 1.1, either Δ_0 or Δ_1 is available to use for f . Here, we assume that Δ_0 is available. The other case is similar. Using Δ_0 , the constraint $g = f^{x_1=0}$ is effectively T-constructible from $\{f, \Delta_0\}$. If $g \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, then the induction hypothesis leads to the desired conclusion. Therefore, we assume below that g is in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$.

(a) Assume that $g \in \mathcal{DG}$. When $g = [u, 0, \dots, 0]$, f has the form $[u, 0, \dots, 0, u]$. This constraint f thus belongs to \mathcal{ED}_1 , a contradiction against $f \notin \mathcal{ED}_1^{(+)}$.

(b) Assume that $g \in \mathcal{ED}_1^{(+)}$. Since $g = [u, 0, \dots, 0, x]$, f is of the form $[u, 0, \dots, 0, x, u]$. Let us define $h = f^{x_1=*}$, which equals $[u, 0, \dots, x, u + x]$. When $u \neq -x$, since $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, we apply the induction hypothesis. On the contrary, if $u = -x$, then we consider $f^2 (= [u^2, 0, \dots, 0, x^2, u^2])$ and reduce this case to the case of $u \neq -x$.

(c) Next, we assume that $g \in \mathcal{AZ}$. In this case, h has the form $g = [u, 0, u, 0, \dots, 0]$; thus, f equals $[u, 0, u, 0, \dots, 0, u] \in \mathcal{AZ}$, a contradiction. Moreover, when $g = [u, 0, u, 0, \dots, u]$, since $f = [u, 0, u, 0, \dots, u, u]$, we apply Δ_0 repeatedly to f and then obtain $g' = [0, u, u]$, which obviously belongs to \mathcal{OR} .

(ii) Let us consider the case where $|u| < |w|$. The other case $|u| > |w|$ is similar and omitted. Here, we define $h'(x_1) = f(x_1, x_1, \dots, x_1)$. Obviously, $h' = [u, w]$ holds and thus h' approximates $[0, 1]$. As a result, we effectively T-construct $g = f^{x_1=1}$ from $\{f, \Delta_1\}$. In the case of $g \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, the desired result follows from the induction hypothesis. Hereafter, we assume that g is indeed in $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$.

(a) If $g \in \mathcal{DG}$, then g must have the form $[0, 0, \dots, 0, w]$; thus, f equals $[u, 0, 0, \dots, 0, w]$. Obviously, f belongs to $\mathcal{ED}_1^{(+)}$, a contradiction.

(b) Consider the case where $g \in \mathcal{ED}_1^{(+)}$. Let g equal $[x, 0, \dots, 0, w]$ for a certain constant $x \neq 0$. Notice that f is of the form $[u, x, 0, \dots, 0, w]$. If $u \neq -x$, then we define $h = f^{x_1=*}$, which is $[u + x, x, 0, \dots, 0, w]$. Since $u + x \neq 0$, h does not belong to $\mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. We then apply the induction hypothesis to obtain the desired result. On the contrary, when $u = -x$, we instead substitute $f^2 = [u^2, x^2, 0, \dots, 0, w^2]$ for f and make this case reduced to the previous case of $u \neq -x$.

(c) Finally, consider the case where $g \in \mathcal{AZ}$. Since g cannot have the form $[0, x, 0, x, \dots, x, 0]$, we may assume that $g = [0, x, 0, x, \dots, x]$. The original f should have the form $f = [u, 0, x, 0, x, \dots, x]$ with $x = w$. Notice that $x \neq u$ because $|u| < |w| = |x|$. Let us define $h = f^{x_1=*} = [u, x, x, \dots, x]$. Since $0 < |u| < |x|$, it holds that $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$. We then apply the induction hypothesis to h .

[Case: $u = w = 0$] By Theorem 1.1, either Δ_0 or Δ_1 is available for f . Here, we assume that Δ_0 is available. With this Δ_0 , we effectively T-construct $g = f^{x_1=0}$ from $\{f, \Delta_0\}$. Note that $g(0, \dots, 0) = 0$. For the following argument, we set $h = f^{x_1=*}$. Note that, since $g \notin \mathcal{ED}_1^{(+)}$, g must be either in \mathcal{DG} or in \mathcal{AZ} .

(a) Assume that g is in \mathcal{DG} . If g is of the form $[0, 0, \dots, 0, x]$ with $x \neq 0$, then f equals $[0, 0, \dots, 0, x, 0]$. It thus follows that $h = [0, 0, \dots, x, x]$. Since $h \notin \mathcal{DG} \cup \mathcal{ED}_1^{(+)} \cup \mathcal{AZ}$, we can apply the induction hypothesis.

(b) When $g \in \mathcal{AZ}$, g must have the form $[0, x, 0, x, \dots, 0]$, implying that $f = [0, x, 0, x, \dots, 0, 0]$. The induction hypothesis then leads to the desired consequence. \square

Appendix: Proof of Lemma 2.1

In what follows, we will give the missing proof of Lemma 2.1. For any constraint f of arity k , the notation $\max |f|$ indicates the maximum value $|f(x)|$ over all values $x \in \{0, 1\}^k$.

(1–2) These properties directly come from the definition of effective T-constructibility.

(3) This claim is split into two parts: $\#\text{CSP}(f, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\Lambda, \mathcal{F})$ and $\#\text{CSP}(\Lambda, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\mathcal{G}, \mathcal{F})$, where Λ is a p-convergence series for f and it is also effectively T-constructible from \mathcal{G} . We will prove these parts separately.

(a) First, we intend to show that $\#\text{CSP}(\Lambda, \mathcal{F}) \leq_{\text{AP}} \#\text{CSP}(\mathcal{G}, \mathcal{F})$. Let $\Lambda = (f_1, f_2, \dots)$ and $\mathcal{G} = \{g_1, g_2, \dots, g_d\}$. Now, we take any constraint frame $\Omega = (G, X | \mathcal{F}', \pi)$ with $\mathcal{F}' \subseteq \Lambda \cup \mathcal{F}$, given to $\#\text{CSP}(\Lambda, \mathcal{F})$. Since the constraint set \mathcal{F}' is finite, for simplicity, we assume that \mathcal{F}' is composed of constraints $h_1, h_2, \dots, h_s, f_{i_1}, f_{i_2}, \dots, f_{i_t}$, where $s \in \mathbb{N}$, $t \in \mathbb{N}^+$, and each constraint h_i belongs to $\mathcal{F} - \Lambda$. For this

constraint frame Ω , we will explain how to compute the value $cs\pi_\Omega$. Since Λ is effectively T-constructible from \mathcal{G} , there exists a polynomial-time DTM M that, for each index $j \in [t]$, generates an appropriate graph \tilde{G}_{i_j} realizing f_{i_j} from any graph G_{i_j} representing f_{i_j} .

Each node v labelled f_{i_j} ($j \in [t]$) in G corresponds to a unique subgraph G_{i_j} , including all dangling edges adjacent to v , that represents f_{i_j} . By running M on G_{i_j} , we obtain another subgraph \tilde{G}_{i_j} realizing f_{i_j} , which contains all the dangling edges of G_{i_j} . It is therefore possible to generate from G another bipartite graph \tilde{G} in which every subgraph G_{i_j} of G representing f_{i_j} is replaced by its associated subgraph \tilde{G}_{i_j} obtained from G_{i_j} by M . We denote by Ω' the constraint frame obtained from Ω by replacing G with \tilde{G} and by modifying π accordingly. The definition of “realizability” implies that $cs\pi_{\Omega'}$ equals $\gamma \cdot cs\pi_\Omega$ for an appropriate number $\gamma \in \mathbb{A}$. Since \tilde{G} contains only constraints in $\mathcal{G} \cup \mathcal{F}$, Ω' must be a valid input instance to $\#CSP(\mathcal{G}, \mathcal{F})$. Thus, we conclude that $\#CSP(\Lambda, \mathcal{F})$ is AP-reducible to $\#CSP(\mathcal{G}, \mathcal{F})$.

(b) Next, we want to claim that $\#CSP(f, \mathcal{F}) \leq_{AP} \#CSP(\Lambda, \mathcal{F})$. This claim is proven by modifying the proof of [11, Lemma 9.2]. Hereafter, assume that f is of arity k and let $\Lambda = (g_1, g_2, \dots)$. For convenience, we define $AC = \{x \in \{0, 1\}^k \mid f(x) \neq 0\}$. By Eq.(2), there exists a constant $\lambda \in (0, 1)$ such that, for every $m \in \mathbb{N}^+$ and every $x \in \{0, 1\}^k$, certain constants $c, d \in \{\pm 1\}$ must satisfy

$$(*) \quad (1 + \lambda^m c)g_m(x) \leq f(x) \leq (1 + \lambda^m d)g_m(x) \text{ for all } x \in AC, \text{ and } |g_m(x)| \leq \lambda^m \text{ for all } x \in \{0, 1\}^k - AC.$$

Without loss of generality, we can assume that γ is an algebraic number.

Let us take any constraint frame $\Omega = (G, X | \mathcal{F}', \pi)$ with $G = (V_1 | V_2, E)$ and $\mathcal{F}' \subseteq \{f\} \cup \mathcal{F}$ given as an input instance to $\#CSP(f, \mathcal{F})$. It is enough to consider the case where f is in \mathcal{F}' . Let L denote the set of all 2^k -tuples $\ell = (\ell_{x_1}, \ell_{x_2}, \dots, \ell_{x_{2^k}}) \in \mathbb{N}^{2^k}$ satisfying that $\sum_{i \in [2^k]} \ell_{x_i} = |V_2|$, where each x_i denotes the lexicographically i th string in $\{0, 1\}^k$. In addition, we set $L_f = \{\ell \in L \mid \forall i \in [2^k] [f(x_i) = 0 \rightarrow \ell_{x_i} = 0]\}$. It is not difficult to show that $cs\pi_\Omega$ is always expressed in the form $\sum_{\ell \in L_f} \alpha_\ell (\prod_{x \in AC} f(x)^{\ell_x})$ for appropriate numbers $\alpha_\ell \in \mathbb{N}$.

We set $a_0 = 2^k! 2^{4k}$ and $b_0 = [1 + (2 \max |f|)^{2^{2k}}] 2^{|X|}$, which are obviously independent of m . Meanwhile, we arbitrarily fix an integer $m \in \mathbb{N}^+$ that satisfies both $\lambda^m a_0 < 1$ and $\lambda^m b_0 < 1$, and we denote by Ω_m the constraint frame obtained from Ω by replacing every node labeled f with a new node having the label g_m . Concerning this Ω_m , its value $cs\pi_{\Omega_m}$ coincides with the sum $\Gamma_{1,m} + \Gamma_{2,m}$, where

$$\Gamma_{1,m} = \sum_{\ell \in L_f} \alpha_\ell \prod_{x \in AC} g_m(x)^{\ell_x} \quad \text{and} \quad \Gamma_{2,m} = \sum_{\ell \in L - L_f} \alpha_\ell \prod_{x \in \{0,1\}^k \wedge \ell_x > 0} g_m(x)^{\ell_x},$$

where each notation α_ℓ for $\ell \in L - L_f$ indicates an appropriately chosen algebraic number. The important property of α_ℓ 's is that $\sum_{\ell \in L} \alpha_\ell$ equals $2^{|V_1|}$ ($= 2^{|X|}$), which is the number of all possible Boolean assignments on X .

We begin with establishing a close relationship between $cs\pi_\Omega$ and $\Gamma_{1,m}$; more specifically, we aim at proving the following claim.

Claim 6 *It holds that $(1 + \lambda^m B)\Gamma_{1,m} \leq cs\pi_\Omega \leq (1 + \lambda^m B')\Gamma_{1,m}$ for appropriate numbers $B, B' \in \mathbb{A}$ satisfying $|B|, |B'| \leq a_0$. Therefore, $sgn(cs\pi_\Omega) = sgn(\Gamma_{1,m})$ holds.*

Proof. We note that the second part of the claim follows immediately from the first part because $\lambda^m |B|, \lambda^m |B'| < 1$ holds for our choice of m . Henceforth, we intend to prove the first part. Fix $\ell \in L_f$ arbitrarily. From Condition (*), for appropriate selections of $c_{\ell,x}$'s and $d_{\ell,x}$'s in $\{\pm 1\}$, we obtain

$$\prod_{x \in AC} (1 + \lambda^m c_{\ell,x})^{\ell_x} g_m(x)^{\ell_x} \leq \prod_{x \in AC} f(x)^{\ell_x} \leq \prod_{x \in AC} (1 + \lambda^m d_{\ell,x})^{\ell_x} g_m(x)^{\ell_x}. \quad (5)$$

Note that, when all elements in \mathcal{F}' are limited to *nonnegative* constraints, we can always set $c_{\ell,x} = -1$ and $d_{\ell,x} = 1$. Eq.(5) leads to upper and lower bounds of $cs\pi_\Omega$:

$$\sum_{\ell \in L_f} \alpha_\ell \prod_{x \in AC} (1 + \lambda^m c_{\ell,x})^{\ell_x} g_m(x)^{\ell_x} \leq cs\pi_\Omega \leq \sum_{\ell \in L_f} \alpha_\ell \prod_{x \in AC} (1 + \lambda^m d_{\ell,x})^{\ell_x} g_m(x)^{\ell_x}. \quad (6)$$

Let us further estimate the first and the last terms in Eq.(6). Let us deal with the first term. By considering the binomial expansion of $(1 + z)^n$, it holds that, for any numbers $n \in \mathbb{N}^+$ and $z \in \mathbb{R}$ satisfying that $-1/n \leq z \leq 2/n$, there exists a number $e \in \{1/2, n\}$ such that $1 + nz \leq (1 + z)^n \leq 1 + enz$ (more precisely,

if $z \geq 0$ then $e = n$; otherwise, $e = 1/2$). Hence, by choosing appropriate numbers $e_{\ell,x} \in \{\pm 1/2, \pm \ell_x\}$, we obtain

$$\prod_{x \in AC} (1 + \lambda^m c_{\ell,x})^{\ell_x} g_m(x)^{\ell_x} \geq \prod_{x \in AC} (1 + \lambda^m \ell_x e_{\ell,x}) g_m(x)^{\ell_x} = \prod_{x \in AC} (1 + \lambda^m \ell_x e_{\ell,x}) \prod_{x \in AC} g_m(x)^{\ell_x}$$

since m satisfies that $-1 < \lambda^m \ell_x e_{\ell,x} < 1$.

For a further estimation, let us focus on the value $\prod_{x \in AC} (1 + \lambda^m z_x)$ for any series $\{z_x\}_{x \in AC} \subseteq [-2^{2k}, 2^{2k}]_{\mathbb{Z}}$. Since $\prod_{x \in AC} (1 + \lambda^m z_x)$ has the form $1 + \sum_{i=1}^{|AC|} \sum_{y_1, y_2, \dots, y_i \in AC} \lambda^{im} z_{y_1} z_{y_2} \cdots z_{y_i}$, where all indices y_1, y_2, \dots, y_i are distinct, if we set $B = \sum_{i=1}^{|AC|} \sum_{y_1, y_2, \dots, y_i \in AC} \lambda^{(i-1)m} |z_{y_1} z_{y_2} \cdots z_{y_i}|$, then we derive that $1 - \lambda^m B \leq \prod_{x \in AC} (1 + \lambda^m z_x) \leq 1 + \lambda^m B$. Note that $|\lambda^m z_{y_j}| \leq 1$ for any $x \in AC$. Therefore, it follows that

$$\sum_{y_1, \dots, y_i \in AC} \lambda^{(i-1)m} |z_{y_1} \cdots z_{y_i}| \leq \sum_{y_1 \in AC} |z_{y_1}| \sum_{y_2, \dots, y_i \in AC} 1 \leq |AC| 2^k \binom{|AC|}{i} \leq |AC| 2^{2k} |AC|!$$

We then conclude that B must satisfy $|B| \leq \sum_{i=1}^{|AC|} |AC| 2^{2k} |AC|! \leq |AC|^{2^{2k}} |AC|! \leq a_0$ because of $|AC| \leq 2^k$. From this fact, an appropriately chosen series $\{B_\ell\}_{\ell \in L_f} \subseteq \mathbb{A}$ with $|B_\ell| \leq a_0$ satisfies that

$$\prod_{x \in AC} (1 + \lambda^m \ell_x e_{\ell,x}) \prod_{x \in AC} g_m(x)^{\ell_x} \geq (1 + \lambda^m B_\ell) \prod_{x \in AC} g_m(x)^{\ell_x}.$$

Finally, we claim the existence of an appropriate number $B \in \mathbb{A}$ with $|B| \leq a_0$ for which

$$\sum_{\ell \in L_f} \alpha_\ell (1 + \lambda^m B_\ell) \prod_{x \in AC} g_m(x)^{\ell_x} \geq (1 + \lambda^m B) \sum_{\ell \in L_f} \alpha_\ell \prod_{x \in AC} g_m(x)^{\ell_x} = (1 + \lambda^m B) \Gamma_{1,m}.$$

For an estimation of the third term in Eq.(6), a similar argument gives the existence of an algebraic number $B' \in \mathbb{A}$ such that $|B'| \leq a_0$ and

$$\sum_{\ell \in L_f} \alpha_\ell \prod_{x \in AC} (1 + \lambda^m d_{\ell,x})^{\ell_x} g_m(x)^{\ell_x} \leq (1 + \lambda^m B') \Gamma_{1,m}.$$

By the selection of B and B' , they certainly satisfy the claim. \square

Next, we will give a bound of $\Gamma_{2,m}$; in particular, we aim at showing the following claim.

Claim 7 *It holds that $|\Gamma_{2,m}| \leq \lambda^m b_0$.*

Proof. In what follows, we assume that $\ell \in L - L_f$. For this ℓ , there exists an element x such that $x \in AC$ and $\ell_x > 0$. For convenience, we define $D = \{x \in \{0, 1\}^k \mid \ell_x > 0\}$ and further partition it into two sets: $D_1 = \{x \in D \mid x \in AC\}$ and $D_2 = \{x \in D \mid x \notin AC\}$. Notice that D_2 is nonempty. Since $|g_m(x)| \leq \lambda^m$ for all $x \in D_2$, it follows that, since $\lambda < 1$,

$$\left| \prod_{x \in D_2} g_m(x)^{\ell_x} \right| = \prod_{x \in D_2} |g_m(x)|^{\ell_x} \leq \prod_{x \in D_2} \lambda^{m \ell_x} \leq \lambda^m.$$

Condition (*) implies that $|g_m(x)| \leq |f(x)| / \min\{1 + \lambda^m c, 1 + \lambda^m d\} \leq 2|f(x)|$ for any $x \in AC$ because $|\lambda^m c|, |\lambda^m d| < 1/2$. If $\max |g_m| \geq 1$, then we obtain

$$\left| \prod_{x \in D_1} g_m(x)^{\ell_x} \right| \leq \prod_{x \in D_1} |g_m(x)|^{\ell_x} \leq (\max |g_m|)^{2^k |AC|} \leq (2 \max |f|)^{2^{2k}}.$$

When $\max |g_m| < 1$, we instead obtain $\prod_{x \in D_1} |g_m(x)|^{\ell_x} \leq 1$. Therefore, it holds that

$$\left| \prod_{x \in D} g_m(x)^{\ell_x} \right| = \left| \prod_{x \in D_2} g_m(x)^{\ell_x} \right| \cdot \left| \prod_{x \in D_1} g_m(x)^{\ell_x} \right| \leq \lambda^m C,$$

where $C = 1 + (2 \max |f|)^{2^{2k}}$. The value $|\Gamma_{2,m}|$ is upper-bounded by

$$|\Gamma_{2,m}| \leq \sum_{\ell \in L - L_f} \alpha_\ell \left| \prod_{x \in D} g_m(x)^{\ell_x} \right| \leq \lambda^m C \sum_{\ell \in L - L_f} \alpha_\ell \leq \lambda^m b_0.$$

This completes the proof of the claim. \square

To finish the proof of Lemma 2.1, we will present a randomized oracle computation of $\#\text{CSP}(f, \mathcal{F})$ that makes a single query to the oracle $\#\text{CSP}(\Lambda, \mathcal{F})$. First, we want to define a special constant d_0 corresponding to Ω . The definition of d_0 requires the following well-known lower bound of the absolute values of polynomials in algebraic numbers.

Lemma 4.5 [10] *Let $\alpha_1, \dots, \alpha_m \in \mathbb{A}$ and let c be the degree of $\mathbb{Q}(\alpha_1, \dots, \alpha_m)/\mathbb{Q}$. There exists a constant $e > 0$ that satisfies the following statement for any complex number α of the form $\sum_k a_k \left(\prod_{i=1}^m \alpha_i^{k_i} \right)$, where $k = (k_1, \dots, k_m)$ ranges over $[N_1] \times \dots \times [N_m]$, $(N_1, \dots, N_m) \in \mathbb{N}^m$, and $a_k \in \mathbb{Z}$. If $\alpha \neq 0$, then $|\alpha| \geq (\sum_k |a_k|)^{1-c} \prod_{i=1}^m e^{-cN_i}$.*

Following closely the proof of [11, Lemma 9.2] under the assumption that $csp_\Omega \neq 0$, it is possible to set values of the series $\{N_i\}_i$, $\{a_k\}_k$, $\{\alpha_i\}_i$, and $\{k_i\}_i$ appropriately so that Lemma 4.5 provides two constants $c, e > 0$ for which $|csp_\Omega| \geq (\sum_k |a_k|)^{1-c} \prod_i e^{-cN_i}$. The desired constant d_0 is now defined to be $(\sum_k |a_k|)^{1-c} \prod_i e^{-cN_i}$. Notice that d_0 is in \mathbb{A} .

Next, we describe our randomized approximation algorithm.

[Algorithm \mathcal{M}] On input instance $(\Omega, 1/\varepsilon)$, set $\delta = \varepsilon/2$ and find an integer $m \geq 1$ in polynomial time satisfying that $\lambda^m a_0 < \min\{1, \delta\}$ and $\lambda^m b_0 < \min\{d_0, \delta\}$. Produce another constraint frame Ω_m . Send a query word $(\Omega_m, 1/\delta)$ to the oracle and let w be an answer from the oracle. Notice that w is a *random variable* since the oracle is a RAS. Compute d_0 defined above. If $|w| < d_0$, then output 0; otherwise, output w .

We claim that the algorithm \mathcal{M} indeed solves $\#\text{CSP}(f, \mathcal{F})$ with high probability. Let us consider two cases separately.

(1) For the first case where $csp_\Omega = 0$, we need to output 0 with high probability. Let us evaluate the values $\Gamma_{1,m}$ and $\Gamma_{2,m}$. Obviously, Claim 6 implies $\Gamma_{1,m} = 0$. By Claim 7 and the choice of m , we derive $|\Gamma_{2,m}| \leq \lambda^m b_0 < d_0$. Thus, $|csp_{\Omega_m}| < d_0$ holds since $csp_{\Omega_m} = \Gamma_{1,m} + \Gamma_{2,m}$. This means that \mathcal{M} outputs 0 with high probability.

(2) Next, we consider the case where $csp_\Omega \neq 0$. Let us assume that $csp_\Omega > 0$. Note that $csp_\Omega \geq d_0$. Next, we choose a number α , not depending on m , for which $\alpha(\Gamma_{1,m} + b_0) \leq B\Gamma_{1,m} - b_0$ and $|\alpha| \leq \max\{a_0, b_0\}$. For this α , it holds that $(1 + \lambda^m \alpha)csp_{\Omega_m} = (1 + \lambda^m \alpha)(\Gamma_{1,m} + \Gamma_{2,m}) \leq (1 + \lambda^m B)\Gamma_{1,m}$. Similarly, we choose α' with $|\alpha'| \leq \max\{a_0, b_0\}$ such that $(1 + \lambda^m B)\Gamma_{1,m} \leq (1 + \lambda^m \alpha')(\Gamma_{1,m} + \Gamma_{2,m}) = (1 + \lambda^m \alpha')csp_{\Omega_m}$.

For simplicity, let $\gamma = \max\{|\alpha|, |\alpha'|\}$. Note that $\delta \geq \lambda^m \gamma$. Since $\lambda^m \gamma < 1$, it holds that $\log_2(1 + \lambda^m \gamma) \leq \lambda^m \gamma \leq \delta$. Thus, we conclude that $1 + \lambda^m \alpha' \leq 2^{\log_2(1 + \lambda^m \gamma)} \leq 2^\delta$. Moreover, since $\log_2(1 - \lambda^m \gamma) \geq -\lambda^m \gamma$, it follows that $1 + \lambda^m \alpha \geq 2^{\log_2(1 - \lambda^m \gamma)} \geq 2^{-\delta}$. In conclusion, it holds that $2^{-\delta}csp_\Omega \leq csp_{\Omega_m} \leq 2^\delta csp_\Omega$. From this follows $csp_{\Omega_m} > 0$.

If w is any oracle answer, then it must satisfy that $2^{-\delta}csp_{\Omega_m} \leq w \leq 2^\delta csp_{\Omega_m}$ because $csp_{\Omega_m} > 0$. Therefore, we derive that $w \leq 2^\delta csp_{\Omega_m} \leq 2^{2\delta}csp_\Omega$ and $w \geq 2^{-\delta}csp_{\Omega_m} \geq 2^{-2\delta}csp_\Omega$. Since $\varepsilon = 2\delta$, \mathcal{M} outputs a 2^ε -approximate solution using any 2^δ -approximate solution for $(\Omega_m, 1/\delta)$. The case $csp_\Omega < 0$ can be similarly handled.

This completes the proof of Lemma 2.1

References

- [1] J. Cai and P. Lu. Holographic algorithms: from arts to science. *J. Comput. System Sci.*, 77 (2011), 41–61.
- [2] J. Cai, P. Lu, M. Xia. Holant problems and counting CSP. In *Proc. of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009)*, pp.715–724, 2009.
- [3] N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *J. Comput. System Sci.*, 51 (1995) 511–522.
- [4] N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Inform. Comput.*, 125 (1996) 1–12.
- [5] V. Dalmau and D. K. Ford. Generalized satisfiability with limited occurrences per variable: a study through Δ -matroid parity. In *Proc. of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS 2003)*, Lecture Notes in Computer Science, Springer-Verlag, Vol.2747, pp.358–367, 2003.
- [6] M. Dyer, L. A. Goldberg, C. Greenhill, M. Jerrum. The relative complexity of approximating counting problems. *Algorithmica*, 38 (2004), 471–500.

- [7] M. Dyer, L. A. Goldberg, and M. Jerrum. The complexity of weighted Boolean $\#$ CSP. *SIAM J. Comput.*, 38 (2009), 1970–1986.
- [8] M. Dyer, L. A. Goldberg, and M. Jerrum. An approximation trichotomy for Boolean $\#$ CSP. *J. Comput. System Sci.*, 76 (2010) 267–277.
- [9] T. J. Schaefer. The complexity of satisfiability problems. In *Proc. of the 10th Annual ACM Symposium on Theory of Computing* (STOC 1978), pp.216–226, 1978.
- [10] K. B. Stolarsky. *Algebraic Numbers and Diophantine Approximations.*, Marcel Dekker, 1974.
- [11] T. Yamakami. Approximate counting for complex-weighted Boolean constraint satisfaction problems. *Informa. Comput.*, 219 (2012) 17–38. An early version appeared in the Proceedings of the 8th Workshop on Approximation and Online Algorithms (WAOA 2010), Lecture Notes in Computer Science, Springer-Verlag, vol.6534, pp.261–272, 2011.
- [12] T. Yamakami. A dichotomy theorem for the approximation complexity of complex-weighted bounded-degree Boolean $\#$ CSPs. *Theor. Comput. Sci.* 447 (2012) 120–135. An extended abstract appeared in the Proceedings of the 4th International Conference on Combinatorial Optimization and Applications (COCO 2010), Lecture Notes in Computer Science, Springer-Verlag, vol.6508 (Part I), pp.285–299, 2010.
- [13] T. Yamakami. Optimization, randomized approximability, and constraint satisfaction problems. In *Proc. of the 22nd International Symposium on Algorithms and Computation* (ISAAC 2011), Lecture Notes in Computer Science, Springer-Verlag, vol.7074, pp.454–463, 2011.
- [14] T. Yamakami. Constant unary constraints and symmetric real-weighted counting CSPs. In *Proc. of the 23rd International Symposium on Algorithms and Computation* (ISAAC 2012), Lecture Notes in Computer Science, Springer-Verlag, vol.7676, pp.237–246, 2012.